ANÁLISIS DE SEGURIDAD DE APLICACIONES MÓVILES NATIVAS PARA EL SISTEMA OPERATIVO ANDROID VERSIÓN JELLY BEAN 4.1.2 EN DISPOSITIVOS MÓVILES SMARTPHONE

ING. PEDRO JULIO COLORADO ÁNGEL
ING. INÍRIDA JEANETH TORRES BAQUERO

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA – UNAD
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
VILLAVICENCIO
2015

ANÁLISIS DE SEGURIDAD DE APLICACIONES MÓVILES NATIVAS PARA EL SISTEMA OPERATIVO ANDROID VERSIÓN JELLY BEAN 4.1.2 EN DISPOSITIVOS MÓVILES SMARTPHONE

ING. PEDRO JULIO COLORADO ÁNGEL
ING. INÍRIDA JEANETH TORRES BAQUERO

Trabajo de grado como requisito para optar al título de:
Especialista en Seguridad Informática

Ing. Gabriel Mauricio Ramírez Villegas
Director

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA – UNAD
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
VILLAVICENCIO
2015

CONTENIDO

pág.

LISTA DE FIGURAS

pág.

LISTA TABLAS

pág.

LISTA DE ANEXOS

pág.

INTRODUCCIÓN

Android se ha convertido en el sistema operativo móvil más utilizado en los últimos años, esta popularidad ha incrementado exponencialmente el número de aplicaciones disponibles en el Play Store superando a otras plataformas móviles ofreciendo un total de 1.21 millones de aplicaciones; esta tendencia conlleva al aumento de casos reportados de ataques informáticos sobre esta plataforma mediante la explotación de vulnerabilidades existentes en muchas de las aplicaciones ofrecidas.

Una cantidad considerable de los ataques reportados buscan obtener información confidencial de los usuarios como las cuentas de correo electrónico, contraseñas, números de tarjetas de crédito, números de teléfono, IMEI del dispositivo, etc., mediante la publicación de aplicaciones “maliciosas” que ofrecen alguna utilidad básica pero que tienen un oscuro propósito en el trasfondo.

Los usuarios de las aplicaciones móviles para el sistema operativo Android no son conscientes del riesgo y por ende no toman medidas mínimas en los temas de seguridad de sus dispositivos móviles pues se observa una tendencia hacia el uso de aplicaciones gratuitas para acceso a redes sociales, juegos, correo, compras, etc., las cuales pueden darle un mal uso a los datos sensibles.

En la presente investigación se realiza un análisis de seguridad informática sobre aplicaciones móviles nativas para el sistema operativo Android Jelly Bean versión 4.1.2., enfocados en los riesgos OWASP Mobile M2 - Almacenamiento de datos inseguro y M3 - Protección insuficiente en la capa de transporte buscando evidenciar las vulnerabilidades que puedan existir.

El proyecto se desarrolla siguiendo los lineamientos definidos en la metodología OWASP Mobile Security Testing mediante la adopción de un plan de pruebas propio, el cual busca identificar hallazgos mediante el reconocimiento de la aplicación, ingeniería inversa, análisis estático y análisis dinámico.

La evaluación de seguridad se llevó a cabo utilizando herramientas de software libre, ampliamente reconocidas en el entorno de seguridad informática y pentesting, sobre dispositivos móviles Smartphone físicos y otro usando emuladores.

1. PLANTEAMIENTO DEL PROBLEMA

El uso de dispositivos móviles de tipo Smartphone se ha incrementado y diversificado considerablemente en los últimos años con el desarrollo de nuevas funcionalidades en áreas de oficina, correo, banca electrónica, entretenimiento, juegos, redes sociales, mensajería entre otros, convirtiendo al Smartphone en el nuevo blanco de los ciberdelincuentes que buscan ponerlos en riesgo y comprometer la seguridad de estos dispositivos para acceder a la información personal y/o bancaria del usuario, robando datos, suplantando la identidad o haciéndolos portadores de malware.

Ventajas como movilidad y conectividad en los Smartphone, sumadas a las múltiples funcionalidades ofrecidas por los proveedores de servicios de telefonía han permitido el aumento de los datos transmitidos mediante tecnologías móviles haciendo que estos dispositivos sean utilizados, más allá del servicio de comunicación de voz, como una herramienta de gestión de información privada y sensible aumentando el riesgo que sean vulnerables a los diferentes tipos de ataques informáticos (malware, virus, troyanos, gusanos, bombas de tiempo, software espía, etc.) generando así pérdida de confidencialidad de la información.

RAMÍREZ[1] relaciona al usuario del dispositivo Smartphone como el primer riesgo y vulnerabilidad que puede existir al no contar con una seguridad apropiada, convirtiéndose en una vulnerabilidad bastante grande, permitiendo a cualquiera acceder a la información para revisarla, modificarla o copiarla, al igual que su falta de precaución en la instalación de aplicaciones y la falta de sistemas antivirus.

El uso inadecuado del Smartphone representa un riesgo, que en la mayoría de los casos alcanza a ser crítico, en especial por la cantidad de datos que se almacena y/o gestiona en ellos. Los riesgos más comunes que se pueden presentar en estos dispositivos son la pérdida o robo del dispositivo, comunicaciones inseguras (Wifi abierta, webs maliciosas, bluetooth abierto, NFC), navegación insegura, malware (virus, troyanos, software espía, etc.), pérdida de datos, aplicaciones no confiables y configuración inadecuada del dispositivo.

La mayoría de los usuarios utilizan Smartphone con sistema operativo Android con las configuraciones de fábrica, ignorando problemas de seguridad propios del sistema operativo o sin contar con medidas de protección mínimas, desconociendo así, el riesgo al cual se encuentra expuesta la seguridad de la información gestionada y el impacto que puede generar sobre la misma el convertirse en una víctima de un ataque móvil, generando la necesidad de proteger de forma adecuada y responsable la información utilizada en estos dispositivos.

[1] RAMÍREZ, Gabriel. Seguridad en aplicaciones móviles. Universidad Nacional Abierta y a Distancia, Palmira, 2013. p. 126.

Los Smartphone han tenido una gran acogida en el mercado, presentándose actualmente las siguientes situaciones:

- Al ser Android el sistema operativo más usado por usuarios de Smartphone, los desarrolladores se han enfocado en la creación de una amplia variedad de aplicaciones nativas ignorando, en muchos casos, las recomendaciones y medidas de seguridad mínimas; es así como muchas de estas aplicaciones nativas son distribuidas en las tiendas oficiales (Play Store), tiendas de aplicaciones alternativas o distribuidas en internet, las cuales pueden ser descargadas e instaladas por los usuarios que no tienen en cuenta prácticas mínimas de seguridad desprotegiendo el sistema operativo y el dispositivo.

- Los ciberdelincuentes se han enfocado en la creación de aplicaciones maliciosas para Smartphone con sistema operativo Android, que son distribuidas en la tienda oficial de aplicaciones (Play Store) u otras tiendas alternativas, las cuales las hacen pasar por herramientas antivirus, herramientas de monitoreo, juegos, navegadores web, clientes de redes sociales y mediante técnicas de ingeniería social (mensajes de texto, emails, etc.) los ciberdelincuentes inducen a los usuarios a descargarlas e instalarlas para explotar las vulnerabilidades del sistema operativo o de las aplicaciones instaladas en el dispositivo, logrando acceder a información sensible o robar datos confidenciales del usuario haciendo que los ciberdelincuentes obtengan ganancias ilegales.

1.2 FORMULACIÓN DEL PROBLEMA

¿Cuál es el grado de seguridad de las aplicaciones móviles nativas para el Sistema Operativo Android versión Jelly Bean 4.1.2 en dispositivos móviles Smartphone frente a los riesgos indicados en el proyecto OWASP Mobile riesgos M2 y M3?

2. JUSTIFICACIÓN

Hoy en día, la mayoría de las personas utilizan dispositivos Smartphone que contienen información de valor sensible, como cuentas de correo electrónico, redes sociales, información de cuentas bancarias, datos de formularios, documentos privados, historial de llamadas, mensajes de texto o voz, libretas de direcciones, chats, calendarios, fotos, etc., usan conexiones bluetooth, servicios de geo localización GPS, se conectan a internet a través de su operador de telefonía móvil (3G/4G) o por redes Wi-fi, instalan aplicaciones de la tienda oficial o alternativa y acceden a través de sus navegadores a variedad de páginas web, exponiéndose a los riesgos y vulnerabilidades asociadas con el dispositivo, la red y los centros de datos; los usuarios desconocen los ataques que se pueden presentar como troyanos, gusanos, virus, programas espías, malware, secuestro de dispositivos, phishing, etc., mediante los cuales el atacante logra el robo o pérdida de información, daño del dispositivo y en ocasiones beneficios económicos ilegales. Estas amenazas de seguridad están latentes y los usuarios no son conscientes de los riesgos a los que están expuestos, ignorando las medidas de seguridad que deben adoptar para protegerse y mitigar el impacto, lo cual puede generar graves consecuencias para la integridad, confidencialidad y disponibilidad de la información que se maneja a través de estos dispositivos, razón por la cual es vital estar consciente de la seguridad de la información de estos dispositivos.

En el informe realizado en 2014 por la empresa de estudios de mercado International Data Corporation[2] el sistema operativo más usado por los fabricantes de Smartphones es Android con un 78,6% de ventas a nivel mundial, consolidándolo como el de más aceptación en el mundo y convirtiéndolo en un blanco atractivo para los creadores de software malicioso. En la figura 1 se relacionan las ventas de Smartphone a usuarios en el año 2014.

Figura 1. Ventas de Smartphone por sistema operativo en 3er periodo año 2014

Operating System	3Q14 Units	3Q14 Market Share (%)	3Q13 Units	3Q13 Market Share (%)
Android	250,060.2	83.1	205,243	82.0
iOS	38,186.6	12.7	30,330	12.1
Windows	9,033.4	3.0	8,916	3.6
Blackberry	2,419.5	0.8	4,401	1.8
Other OS	1,310.2	0.4	1,407	0.6
Total	**301,009.9**	**100.0**	**250,296.8**	**100.0**

Fuente Gartner, Inc. [En línea]. http://www.gartner.com/newsroom/id/2944819

[2] INTERNATIONAL DATA CORPORATION IDC. Android and iOS Continue to Dominate the Worldwide Smartphone Market with Android Shipments Just Shy of 800 Million in 2013. 2014. [en línea] [citado el 20 Mayo, 2014]. Disponible en internet <http://www.idc.com/getdoc.jsp?containerId=prUS24676414>

Los boletines reportados por Kaspersky Lab[3] confirman el aumento de ataques en el sistema operativo Android y el alto crecimiento de aplicaciones de tipo malware, troyanos y backdoors que se están masificando en la red, mostrando como cifras de ataques a Smartphone[4] que el 33,5% tenía como objetivo el robo de dinero de los usuarios, el 20,6% fue el robo de datos y el 19,4% ganancias en dinero.

Según un amplio estudio solicitado al Global Privacy Enforcement Network (Red Global de Control de Privacidad) las prácticas de privacidad de los creadores de aplicaciones dejan mucho que desear: de 1.211 aplicaciones analizadas, el 75% solicitan permisos para acceder a las funciones del Smartphone o a los datos del usuario, además, muchas de ellas tienen fallas para comunicar al usuario su política de privacidad de una forma transparente y de fácil comprensión; en el 59% de los casos, el creador de la aplicación, previo a la instalación, no logró informar al usuario sobre la recolección de datos, y casi un tercio de las aplicaciones incluidas en la muestra solicitó permiso para acceder a la información irrelevante para la funcionalidad de las aplicaciones; al final, un 15% de las aplicaciones examinadas resultaron ser suficientemente transparente sobre qué información podría accederse y cómo sería utilizada.

Estos informes y estadísticas nos demuestran el aumento considerable de los riesgos y amenazas a los cuales están expuestos los usuarios de Android, así como las vulnerabilidades que pueden contener las aplicaciones desarrolladas para este sistema operativo, razón por la cual la presente investigación pretende realizar un análisis de seguridad a nivel de aplicaciones móviles nativas para el sistema operativo Android en dispositivos móviles Smartphone, con el fin de identificar y documentar las posibles vulnerabilidades que se puedan encontrar en las aplicaciones analizadas buscando generar conciencia en los usuarios sobre los riesgos que implica el uso de las aplicaciones, y en los desarrolladores para fomentar la implementación de buenas prácticas para la seguridad de la información.

[3] KASPERSKY LAB. *Kaspersky Security Bulletin 2013. Overall statistics for 2013.* Diciembre de 2013. [en línea] [citado el 25 Mayo, 2014]. Disponible en internet: <https://www.securelist.com/en/analysis/204792318/Kaspersky_Security_Bulletin_2013_Overall_statistics_for_2013>

[4] KASPERSKY LAB. Abril de 2014. ¿Por qué quieren los cibercriminales tu Smartphone?. [en línea] [citado el 25 Mayo, 2014]. Disponible en internet: <http://newsroom.kaspersky.eu/fileadmin/user_upload/es/Downloads/Kaspersky_pressrelease_Por_qu%C3%A9_quieren_los_cibercriminales_tu_smartphone.doc.pdf> p. 1.

3. OBJETIVOS

3.1 GENERAL

Realizar un análisis de seguridad informática enfocado en los riesgos OWASP Mobile M2 y M3 sobre aplicaciones móviles nativas para el sistema operativo Android, versión Jelly Bean 4.1.2, para identificar vulnerabilidades que puedan afectar la seguridad de la información del dispositivo móvil.

3.2 ESPECÍFICOS

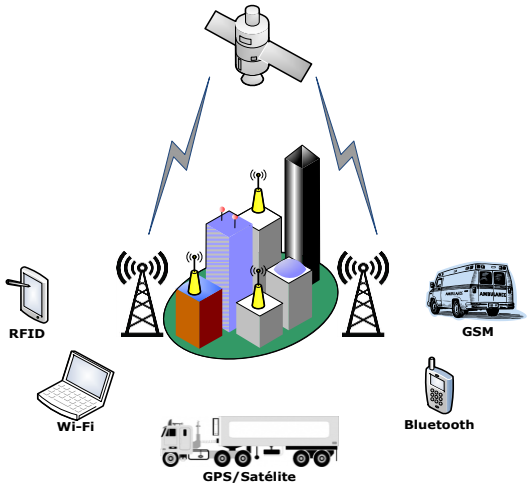
- Esquematizar la arquitectura del sistema operativo Android y el modelo de seguridad enfocado a las aplicaciones móviles nativas.
- Identificar los principales vectores de ataques definidos en OWASP Mobile para los riesgos M2 y M3.
- Realizar evaluación de seguridad informática a aplicaciones móviles nativas siguiendo la metodología OWASP Mobile Security Testing.
- Diagnosticar y presentar el estado de seguridad de las aplicaciones móviles nativas evaluadas.

4. MARCO REFERENCIAL

4.1 MARCO TEÓRICO

4.1.1 Computación Móvil. Comunicación de diferentes equipos portátiles o móviles de hardware y software, que hacen uso de la computación para realizar diferentes tareas computacionales permitiendo la movilidad y la conexión a otros dispositivos por medio de diferentes tecnologías de comunicación inalámbrica y en la administración de forma óptima del procesamiento, almacenamiento y el consumo de la energía. Entre los dispositivos móviles se encuentran actualmente los computadores portátiles, minicomputadores, teléfonos celulares, Smartphone, Tablets, e-Readers, etc., en general cualquier dispositivo que tenga y permita la conexión a otros dispositivos por medio de diferentes tecnologías de comunicación inalámbrica (Wi-Fi (Wireless Fidelity), GSM (Global System for Mobile), Bluetooth, RFID (Radio Frequency Identification), GPRS (General Packet Radio Service) y Satelital)[5] [6] [7]. En la figura 2 se presenta un ejemplo de computación móvil.

Figura 2. Ejemplo Computación móvil



Fuente: Los autores

Entre las principales ventajas de la computación móvil se encuentra la movilidad y flexibilidad que permite enviar y recibir información sin los obstáculos o

[5] RAMÍREZ, Gabriel. La importancia de la computación móvil: pasado, presente y futuro. Revista Especializada en Telecomunicaciones, Electrónica y Sistemas. Universidad Nacional Abierta y a Distancia. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet: <http://datateca.unad.edu.co/contenidos/201493/Articulos/articulo_6-6.pdf>. Volumen 2, Número 2. p. 3.
[6] ASOCIACIÓN COLOMBIANA DE INGENIEROS DE SISTEMAS Y MUSEO COLOMBIANO DE INFORMÁTICA, E IBM. Historia de la Computación Historia de la Computación. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet: <http://www.acis.org.co/archivosAcis/HistoriadelaComputacion.pdf>
[7] WEISER, Mark. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet:<http://www.ubiq.com/hypertext/>

restricciones de un espacio físico (oficina, campus, edificio), alta escalabilidad para configurar una amplia variedad de topologías de red, facilidad de instalación, bajos costos de implementación, amplia la capacidad de toma de decisiones porque permite obtener y analizar datos críticos, incrementa la productividad y aumenta la cercanía con los clientes de negocio de una organización.

Igualmente, la computación móvil tiene asociadas unas limitaciones, entre las cuales podemos mencionar los dispositivos con recursos limitados para almacenamiento y procesamiento, anchos de banda menores comparado con las redes fijas, problemas de comunicación provocados por cambios en condiciones del ambiente, variación de la intensidad de la señal, riesgos de seguridad asociados a los dispositivos y/o señales emitidas, entre otros.

La computación móvil tiene una variedad de aplicaciones y actualmente se usa bastante en los medios de transporte para establecer la ubicación vía GPS, mantener actualizada la información de condiciones ambientales, estado de vías, noticias, mensajes con vehículos cercanos mediante redes sociales especializadas y comunicación personal; también se utiliza bastante en el campo de salud, donde los funcionarios médicos utilizando dispositivos inalámbricos obtienen y comparten información del estado de los pacientes; en el campo de los negocios se usa bastante esta tecnología pues permite el acceso a información de oficinas, ventas, inventarios, compras de productos entre otro.

4.1.2 Evolución de la Computación Móvil. La computación móvil se desarrolla con el nacimiento de la necesidad de transportar la información, inicialmente solo se trabajaba con computadores centralizados y la información estaba almacenada en un solo lugar, cuando se necesitaba se debía acudir físicamente al lugar donde se encontraban los computadores. Esto funcionaba correctamente pero con el aumento de la información y los usuarios, los centros se hicieron insuficientes para atender la cantidad de peticiones de información solicitadas por los usuarios[8].

Con el tiempo aparecieron los equipos personales conocidos como los equipos de escritorios, se empezaron a utilizar las redes de computadores, los cuales facilitaron el trabajo de las personas y la centralización de la información disminuyo[9].

Un hito importante para la computación móvil fue la aparición de las redes, específicamente las redes inalámbricas, las cuales con su evolución permitían el uso de los equipos de cómputo conectarse a una red sin necesidad de estar

[8] RAMÍREZ, Gabriel. Seguridad en aplicaciones móviles. Contenido didáctico. Universidad Nacional Abierta y a Distancia. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet: <http://datateca.unad.edu.co/contenidos/201493/CONTENIDO%20DIDACTICO%20EXE1/leccion_3_evolucion_de_la_computacion_movil.html>.
[9] Ibíd.

cableada o modificar la estructura de la red, ofreciendo conveniencia y flexibilidad[10].

A finales de los años 40 se estaba desarrollando los primeros sistemas de telefonía móvil que eran radios analógicos que utilizaban inicialmente modulación en amplitud (AM) y posteriormente modulación en frecuencia (FM), con el avance de la época se logró el aumento a frecuencias superiores a los 900 Mhz, aumentando la posibilidad de dar servicio a un mayor número de usuarios y avanzar en la portabilidad de los terminales; en la década de los 90 la segunda generación (2G) tiene como piedra angular la digitalización de las comunicaciones ofreciendo mejor calidad de voz, aumentando el nivel de seguridad y simplificando la fabricación del terminal e implementándose el estándar GSM: Global System for Mobile communications; con el transcurrir del tiempo surge la necesidad de aumentar la capacidad de transmisión de datos para poder ofrecer servicios como la conexión a Internet desde el móvil, la videoconferencia, la televisión y la descarga de archivos naciendo la Tercera Generación (3G) posibilitándose un sistema totalmente nuevo: UMTS (Universal Mobile Telecommunications System) hasta llegar a la generación actual (4G) que ofrece una telefonía móvil un mayor ancho de banda permitiendo, entre muchas otras cosas, la recepción de televisión en Alta Definición[11].

La construcción de equipos de comunicación y teléfonos celulares posibilitó mejoras en las comunicaciones, permitiendo el uso de estas características para convertirse en servicios para los usuarios, hasta llegar a lo que actualmente se denomina como tecnologías de última generación en cuanto a redes de comunicación, el uso de equipos de comunicación inteligente y de servicios de internet sobre las redes de comunicación celular, es decir, la unión de la red de redes –internet- con las redes celulares[12].

4.1.3 Computación Ubicua. WEISER, padre de la computación ubicua en 1991 en su trabajo "*The computer for the Twenty-First Century*", preveía un futuro en el que los ordenadores se han vuelto "invisibles" y se encuentran integrados en todos los objetos que nos rodean a diario, esto es a lo que se denomina computación ubicua.

La ubicuidad se fundamenta en que la computación pueda estar presente en cualquier momento y en cualquier lugar. Para ello hay que dotar de capacidad de computación a casi cualquier objeto y que dicha capacidad de computación debe ser invisible de cara al usuario. De esta forma el usuario estará utilizando dichos elementos computacionales pero sin ser realmente consciente de que para

[10] TALUKDER, Asoke y YAVAGAL, Roopa. Mobile computing. Technology, Applications and Service creation. Editorial Tata McGraw Hill. 2005. ISBN-13: 978-0-07-058807-3.
[11] Ibíd.
[12] RAMÍREZ, Gabriel. La importancia de la computación móvil: pasado, presente y futuro, Op. Cit. p. 5.

realizar una tarea los estará usando Otro aspecto relevante de la computación ubicua es que lo realmente importante es el usuario y sus necesidades.

En el texto Weiser escribió dos bases fundamentales: El sistema distribuido y la computación móvil, ambos sistemas funcionaban sobre cuatro cimientos: el uso inteligente de espacios eficaces, invisibilidad, escala local y ocultación de los desniveles de acondicionamiento[13] [14].

4.1.4 Smartphone. Un "Smartphone" (teléfono inteligente en español) es un dispositivo electrónico que funciona como un teléfono móvil con características similares a las de un computador personal. Estos dispositivos permiten hacer llamadas y enviar mensajes de texto como los teléfonos móviles convencionales, tienen un sistema operativo diseñado para dispositivos enfocados a su uso con redes de telefonía móvil, además incluye funciones y capacidades similares a las de un computador clásico. Una ventaja importante de los Smartphone es que permiten instalar programas o aplicaciones, para diferentes fines, buscando aumentar el procesamiento de datos y la conectividad del usuario, las cuales pueden ser desarrolladas por el fabricante del dispositivo o por un tercero[15].

Las características sobresalientes de los Smartphones son las pantallas táctiles, el sistema operativo, la conectividad a Internet, el acceso al correo electrónico, la agenda, las cámaras integradas, la administración de contactos, el software multimedia para reproducción de música, visualización de fotos, videos y algunos programas de navegación, así como la habilidad de leer documentos de negocios en variedad de formatos como PDF y Microsoft Office[16].

4.1.5 Historia de los Smartphones. El primer Smartphone "Simon" fue creado por IBM y BELLSOUTH en el año 1992, fue un dispositivo que tenía funcionalidades extra como mail, fax, calendario, calculadora e incluso lector de tarjetas PCMCIA[17].

En el año 1996 Nokia lanzó el "9000", era una fusión entre una PDA y un teléfono inalámbrico. Otros modelos de este fabricante incluyeron características como pantalla a color y conectividad Wi-Fi entre otros. El "9210" Communicator, lanzado en 1998 fue el primero en adoptar el sistema operativo SymbianOS[18].

[13] ASOCIACIÓN COLOMBIANA DE INGENIEROS DE SISTEMAS Y MUSEO COLOMBIANO DE INFORMÁTICA, E IBM. Op. cit.
[14] WEISER. Op. cit.
[15] BAZ, Arturo, et al. Dispositivos móviles. Universidad de Oviedo. [en línea] [citado el 25 Mayo, 2014]. Disponible en internet: <http://156.35.151.9/~smi/5tm/09trabajos-sistemas/1/Memoria.pdf>. p. 23.
[16] Ibid.
[17] CONSEJO NACIONAL CONSULTIVO DE CYBER-SEGURIDAD. Malware en smartphones. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: <http://www.bdigital.org/Documents/Malware_Smartphones.pdf>. p. 4.
[18] Ibíd. p. 5.

En el año 1997 Ericsson desarrollo el Ericsson GS88, apodado 'Pamela', disponía del sistema operativo de 16 bit GEOS de GeoWorks, el mismo que se adoptó en los Nokia 9000/9110, traía correo electrónico POP3, SMS, reloj mundial, navegador, modo de vuelo, manos libres integrado, módem integrado, puerto de infrarrojos, conexión al pc por medio de RS232[19] y teclado QWERTY[20].

A partir del año 2000, sobresalen los dispositivos Windows CE como ordenador de bolsillo y especialmente la comercialización del primer modelo de Blackberry con funcionalidades de Smartphone en el año 2002 por parte del fabricante RIM (Research in Motion), modelo optimizado para la gestión del correo electrónico[21].

En el año 2007 Apple Inc., introduce su primera generación de dispositivos iPhone, que sería los primeros que permitían manipularse íntegramente desde su pantalla táctil. Durante estos últimos años, han salido nuevas versiones de iPhone mejorando características como pantalla y el asistente de voz personal SIRI[22].

En el 2008 sale a la luz Android, se convierte en estandarte del consorcio Open Handset Alliance. El primer dispositivo en utilizar Android fue el HTC Dream G1, distribuido por T-Mobile, el cual incluía la integración de aplicaciones de Google (Maps, Calendar, Gmail y Chrome) y el uso de aplicaciones de terceras partes (gratuitas y de pago) mediante la comunidad Android Market[23].

En enero de 2010, Google lanzó al mercado su dispositivo Nexus One basado en Android OS versión 2.2, el cual permitía un manejo sencillo de Android y el gigante coreano SAMSUNG[24] puso en marcha la línea Galaxy S basado en Android, que durante los últimos años han salido nuevas versiones mejorado características de pantalla, cámara, procesador, bluetooth, lector de huellas, resistencia al agua y al polvo y teclado entre otras.

4.1.6 Sistemas Distribuidos. Las aplicaciones software de los dispositivos móviles deben adaptarse a restricciones de memoria, procesamiento, comunicación intermitente y calidad cambiante. El paradigma de computación distribuida que mejor se adapta a estas características se denomina Agente[25] y de esta manera es que se desarrollan los sistemas de computación móvil y ubicua.

[19] Conocido popularmente como cable serie.
[20] MONTOYA, Juan. El Smartphone innovado. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: <https://sites.google.com/site/elsmartphoneinnovando/historia-del-smartphone>
[21] CONSEJO NACIONAL CONSULTIVO DE CYBER-SEGURIDAD. Op. Cit. p. 6.
[22] IPHONE WORLD. Apple quiere que todos los dispositivos integren Siri. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: <http://www.iphoneworld.com.es/2012/01/apple-quiere-que-todos-los-dispositivos.html>
[23] GARZÓN, Juan. La evolución de los celulares Samsung Galaxy S: tercera parte. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: <http://www.cnet.com/es/noticias/samsung-galaxy-s5-evolucion/>
[24] HILL, Simón. A history of Samsung's Galaxy pones and tablets, from the S1 to the S4. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: <http://www.digitaltrends.com/mobile/history-of-samsungs-galaxy-phones-and-tablets/>
[25] Ente que posee la habilidad, capacidad y autorización para actuar en nombre de otro.

De acuerdo a lo anterior es que han evolucionado las plataformas de agentes móviles en dispositivos limitados, y de la mano de servicios básicos para los entornos móviles como son la comunicación, el descubrimiento y anuncio y la seguridad[26].

La utilización de la tecnología de agentes distribuidos permite adaptarse a limitaciones de los equipos para proporcionar mejores servicios a los usuarios finales y mejorar las prestaciones de la red, debido a que los agentes[27]:

- Proporcionan un servicio, pueden enviarse dinámicamente y bajo demanda a los propios usuarios.
- Realizan distribuciones de tareas para realizar actividades de gestión, siendo los propios agentes quienes recopilen los datos y los procesen localmente en la parte del dispositivo móvil.
- Permite que se realicen tareas de forma asíncrona.
- Realizan gran parte del procesamiento de forma local.
- Permiten una mayor independencia de la disponibilidad de la red, ya que su capacidad de movilidad les permite migrar a otros nodos de la red.

Algunas de las aplicaciones de la tecnología de sistemas distribuidos en sistemas de telefonía móvil de tercera generación son tareas de gestión de red y desarrollo del Entorno de Hogar Virtual VHE (Virtual Home Environment[28]).

Además de los beneficios indicados como parte de un sistema de telefonía móvil, un dispositivo con una plataforma de agentes puede proporcionar al usuario mayores servicios de valor añadido si se integra como parte de entornos de computación ubicua. El usuario desde su dispositivo móvil puede controlar su entorno (intensidad de luces, el aire acondicionado, etc.) como si fuera un mando a distancia universal que se autoconfigura según el ambiente en el que se encuentre. También es posible beneficiarse de otros servicios que ofrezcan los dispositivos que están en su entorno más próximo (ej: enviar documentos a la impresora, enviar diapositivas al portátil que tiene conectado el VideoBeam, etc.[29])

4.1.7 Sistema Operativo. Un Sistema Operativo gestiona los recursos del sistema, optimiza su uso y resuelve conflictos. El sistema operativo va a coordinar todo el funcionamiento del hardware, iniciando todos los elementos para que estén preparados para recibir trabajo, va a ordenar cuándo y cómo debe trabajar el hardware. Es el sistema operativo el que va a asignar los recursos hardware a los distintos programas, va a coordinar y llevar el seguimiento de la ejecución de

[26] GRANADOS, Gerardo. Sistemas Distribuidos. Universidad Nacional Abierta y a Distancia. [en línea] [citado el 18 Octubre, 2014]. Disponible en internet: <http://datateca.unad.edu.co/contenidos/208017/ContLin/leccin_3_comunicacin_en_los_sistemas_distribuidos.html>.
[27] Ibíd.
[28] Ibíd.
[29] Ibíd.

todos los programas en el sistema, va a tomar las decisiones para evitar que se produzcan conflictos entre ellos y va a tratar que el sistema sea lo más eficiente[30].

4.1.8 Sistemas Operativos para Dispositivos Móviles. Un sistema operativo para dispositivos móviles es considerado el programa principal y es capaz de administrar todos sus recursos para ser utilizados de manera eficiente, cómoda y sin interrupciones, de tal manera que el usuario pueda mantener una comunicación sin problema haciendo uso de los recursos que el hardware le suministra[31].

Las características más relevantes de un sistema operativo móvil son[32]:

- Kernel Unificado
- Construido por Capas
- Multiproceso y Multitarea.
- Soporte a diferentes Pantallas
- Soporte Multilenguaje
- Multihilo
- Conectividad Inalámbrica
- Administración del Hardware
- Administración de Aplicaciones
- Navegación Web
- Capacidad de Adaptación
- Reinvención y Mejoramiento
- Personalizable
- Multiusuario
- Inteligente

Actualmente los sistemas operativos para dispositivos móviles con mayor participación en el mercado mundial son Android de Google, iOS de Apple, Windows Phone de Microsoft y BlackBerry, Firefox, Tizen[33], los cuales se describen en la tabla 1. Estos sistemas operativos han proporcionado mejoras y nuevas funcionalidades a los Smartphones enfocados en la experiencia del usuario, en impulsar el desarrollo de nuevas aplicaciones y servicios y ofreciendo

[30] CANDELA, Santiago, et al. Fundamentos de sistemas operativos: teoría y ejercicios resueltos. [en línea]. [citado el 16 de mayo, 2014]. Disponible en internet <http://books.google.es/books?id=fRK3lbTrNy4C&dq=sistemas+operativos&source=gbs_navlinks_s> p.4.

[31] FIGUEREDO, Oscar. Sistemas Operativos para Dispositivos Móviles. Entérese, 74-78. 2006, citado por POLANCO, Kristel y BEAUPERTHUY, José. "Android" el sistema operativo de google para dispositivos móviles, Revista Científica Electrónica Ciencias Gerenciales. [en línea]. [citado el 16 de mayo, 2014]. Disponible en internet: <http://www.revistanegotium.org.ve/pdf/19/art4.pdf> p. 81

[32] RAMÍREZ, Gabriel. Seguridad en aplicaciones móviles, Op. Cit. <http://datateca.unad.edu.co/contenidos/233016/EXE_SAM/leccin_1_sistemas_operativos_moviles.html>

[33] GARTNER. Gartner Says Smartphone Sales Grew 46.5 Percent in Second Quarter of 2013 and Exceeded Feature Phone Sales for First Time. [en línea] [citado el 16 mayo, 2014] Disponible en internet: <http://www.gartner.com/newsroom/id/2573415>

diferentes funcionalidades, entornos de trabajo y características de usabilidad en cuanto a la interfaz de usuario, que al final son estos últimos quienes determinan cuál se adapta a sus necesidades.

Tabla 1. Sistemas operativos con mayor participación

Sistema Operativo	Descripción
Android	Plataforma móvil de código abierto disponible libremente para quien desee utilizarlo. En 2008 con la unión a la *Open Handset Alliance (OHA)* Android fue lanzado bajo la licencia de código abierto, permitiendo a los fabricantes de dispositivos personalizar y permitir nuevas experiencias de usuario, impulsar la innovación y elección del consumidor. Tiene como base el kernel del sistema operativo Linux[34]. Entre las versiones más conocidas se encuentran Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean y Kitkat[35]. Es el sistema operativo móvil con mayor crecimiento en el mercado. Se ha convertido en el favorito de los consumidores y desarrolladores, impulsando un fuerte incremento en el consumo de aplicaciones. Mensualmente los usuarios de Android descargan más de 1.5 millones de aplicaciones y juegos de Google Play[36].
Apple iOS	Desarrollado por la compañía Apple para para aprovechar al máximo la avanzada tecnología del hardware (iPhone, iPad e iPod)[37], para lo cual fue diseñado en base a una variante del kernel de MacOS X.
Windows Phone	Desarrollado por Microsoft Corporation para dispositivos móviles, tiene una plataforma de comunicación convergente para las aplicaciones existentes y como ventaja las características compartidas con Microsoft Windows[38].
BlackBerry	Desarrollado por la compañía Research in Motion Limited (actualmente BlackBerry Limited)[39], para distribuirlo con sus equipos Smartphone. Este sistema operativo ofrece las soluciones de movilidad empresarial seguras e integradas.

[34] GUNASEKERA, Sheran . Android Apps Security.Chapter 1: Android Architecture. Editorial Apress. p. 1.
[35] ANDROID. The Android Story. [en línea] [citado el 18 octubre, 2014] Disponible en internet <http://www.android.com/history/>
[36] ANDROID DEVELOPERS [en línea] [citado el 18 octubre, 2014]. Disponible en internet: <http://developer.android.com/about/index.html>
[37] APPLE. iOS 8. El sistema operativo móvil más avanzado del mundo y de qué manera. [en línea] [citado el 18 octubre, 2014] Disponible en internet <https://www.apple.com/es/ios/what-is/>
[38] MICROSOFT CORPORATION. Windows Phone 8 update history. [en línea] [citado el 18 octubre, 2014] Disponible en internet: <http://www.windowsphone.com/en-us/how-to/wp8/basics/windows-phone-8-update-history>
[39] BLACKBERRY LIMITED. What is the BlackBerry Family?. [en línea] [citado el 18 octubre, 2014] Disponible en internet: <http://blogs.blackberry.com/2014/09/what-is-the-blackberry-family/>

Sistema Operativo	Descripción
Firefox OS	Desarrollado por Mozilla para teléfonos inteligentes, su código es abierto y fue creado completamente utilizando HTML5 y otros estándares web abiertos, que lo hace libre de las normas y restricciones de las plataformas privadas existentes[40].
Tizen	Patrocinado por Linux Foundation y la Asociación Tizen, siendo un sistema operativo abierto y flexible construido desde cero para hacer frente a las necesidades de todos los actores del ecosistema de dispositivos móviles, incluidos los fabricantes de dispositivos, operadores móviles, desarrolladores de aplicaciones y proveedores de software independientes

4.1.9 Aplicación Móvil. Enfocando el concepto en el área de la computación móvil, las aplicaciones móviles son los conjuntos de instrucciones lógicas, procedimientos, reglas, documentación, datos e información asociada a estas que funcionan específicamente en dispositivos móviles, como por ejemplo teléfonos inteligentes, televisores inteligentes, tabletas, entre otros.[41] Este tipo de aplicaciones se desarrollan teniendo en cuenta las limitaciones de los propios dispositivos, como por ejemplo el bajo poder de cómputo, la escasa capacidad de almacenamiento, ancho de banda limitado, etc.[42].

Las aplicaciones móviles se pueden clasificar:

a) De acuerdo al mercado para las que han sido desarrolladas: han sido diseñadas para dispositivos móviles específicos, como por ejemplo teléfonos inteligentes, tabletas, televisores inteligentes, reloj, neveras, gafas entre muchos otros dispositivos o aplicaciones que pueden funcionar en todos los dispositivos móviles.

b) De acuerdo al lenguaje de programación en que ha sido desarrollada: se han desarrollado bajo los lenguajes de programación como Java, Objetive C, Bada, WebOS, C#, C++, HTML5, HTML/CSS/JavaScript, entre muchos otros[43]. En la figura 3 se muestra los lenguajes más usados para aplicaciones móviles.

[40] MOZILLA. Faq de Firefox OS. Acerca de Firefox Os. [en línea] [citado el 18 octubre, 2014] Disponible en internet:<https://www.mozilla.org/es-ES/firefox/os/faq/>

[41] STANLEY Morgan. The mobile Internet Report. [en línea] [citado el 18 octubre, 2014] Disponible en internet: <http://www.morganstanley.com/institutional/techresearch/pdfs/2SETUP_12142009_RI.pdf>

[42] ENRÍQUEZ, Juan y CASAS, Sandra. Usabilidad en aplicaciones móviles. Informes Científicos y Técnicos. Publicaciones de actualización continua. Universidad Nacional de la Patagonia Austral [en línea] [citado el 18 octubre, 2014] Disponible en internet http://ict.unpa.edu.ar/files/ICT-UNPA-62-2013.pdf p. 11.

[43] VISIONMOBILE. Report Mobile Megatrends 2012. Citado por RAMIREZ, Gabriel. La seguridad en aplicaciones móviles: estrategias en el mundo actual, Artículo. [en línea]. [citado el 18 de octubre, 2014]. Disponible en internet: <http://datateca.unad.edu.co/contenidos/233016/Articulos/La_Seguridad_en_Aplicaciones_Moviles_Estrategias_en_el_Mundo_Actual_Gabriel_Ramirez.pdf>. p. 1.

Figura 3. Lenguajes más usados para aplicaciones móviles

Fuente: Visión Mobile. [en línea]. Disponible en: http://www.visionmobile.com/product/mobile-megatrends/

c) De acuerdo a la plataforma: aplicaciones móviles desarrolladas en el lenguaje de programación oficial definido por la empresa u organización: Android-Java, iOS-Objetive C, BlackBerry OS-Java, Bada-C++, Windows Phone-C#, Windows 8-C#-C++, WebOS-HTML5-C++, Mobile Web- HTML5-HTML/CSS/JavaScript, Ubuntu OS-HTML5, entre otros.

d) Según la tienda de aplicaciones, ya sea Google Play, App Store, OVI, App Phone Marketplace, App World, entre otras.

e) Según su propósito: basada en el uso y contexto de la aplicación, la forma como se presenta al usuario y como este puede aprovecharla. Podemos encontrar aplicaciones de Medios, mensajes SMS, web site móviles, widgets web móviles, juegos, utilidades, informativas, localización y productividad[44].

f) Según su desarrollo: web, nativas e hibridas. Estos tipos se diferencian en cómo son desarrolladas, lo que pueden hacer, cómo funcionan y cómo se distribuyen[45].

- Aplicación web: Los dispositivos móviles incluyen navegadores web completamente funcionales, por lo que es posible acceder desde ellos a cualquier sitio web al que pueda accederse desde un ordenador. Las aplicaciones web diseñadas para dispositivos móviles emplean los mismos componentes que las aplicaciones web tradicionales y acceden a los mismos datos a través de los mismos servidores. La única diferencia importante entre las aplicaciones web diseñadas para ordenadores

[44] FLING, Brian. Mobile Design and Development.Capitulo 6. Editorial O'Relly. ISBN 978-0-596-15544-5.
[45] IBM CORPORATION. Cómo garantizar la seguridad de las aplicaciones para dispositivos móviles. Software Group, 2012. p. 3-4.

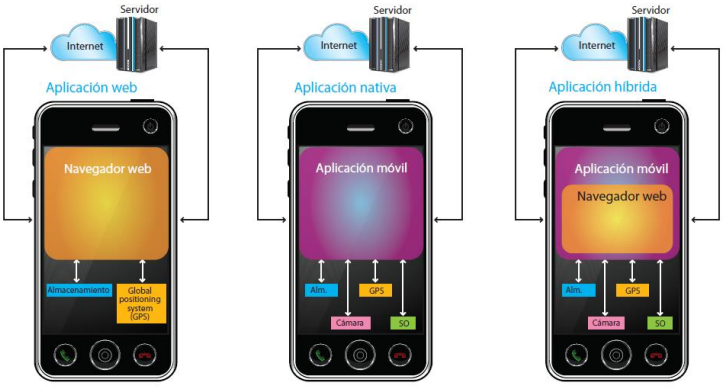
estándar y aquellas diseñadas para dispositivos móviles es cómo son representadas.

La principal ventaja con respecto a la aplicación nativa es la posibilidad de programar independiente del sistema operativo en el que se usará la aplicación. De esta forma se pueden ejecutar en diferentes dispositivos sin tener que crear varias aplicaciones. Las aplicaciones web se ejecutan dentro del propio navegador web del dispositivo a través de una URL.

- Aplicación nativa: son aplicaciones descargadas y ejecutadas en dispositivos móviles. desarrolladas de forma específica para un tipo de dispositivo y su sistema operativo, se basan en la instalación de código ejecutable en el dispositivo del usuario. Tienen la ventaja de acceder a las funciones del dispositivo, como por ejemplo: almacenamiento, GPS (sistema de posicionamiento global), SMS (servicio de mensajes cortos), mails, etc[46]. Ofrecen generalmente mejor rendimiento que las aplicaciones web ejecutadas en navegadores móviles y están mejor integradas con el hardware disponible.

- Aplicación híbrida: son aplicaciones que contienen componentes de navegador web que cargan y ejecutan aplicaciones web. Son un compromiso entre una aplicación web y una aplicación nativa. Con las aplicaciones híbridas, los desarrolladores pueden utilizar componentes de aplicación nativa para personalizar el aspecto y el manejo de la aplicación y componentes de aplicación web para ayudar a superar las limitaciones de actualización de las aplicaciones nativas.

Figura 4. Clasificación aplicaciones móviles



Fuente: IBM Corporation. Cómo garantizar la seguridad de las aplicaciones para dispositivos móviles

[46] SAMBASIVAN, D., et al., Generic Framework for Mobile Application Development. The Second Asian Himalayas International Conference on Internet. Citado por ENRIQUEZ, Juan y CASAS, Sandra. Usabilidad en aplicaciones móviles. [en línea]. [citado el 18 de octubre, 2014]. Disponible en internet: <http://ict.unpa.edu.ar/files/ICT-UNPA-62-2013.pdf>. p. 11-12.

4.1.10 Ecosistema móvil. Plataforma común que permite a una variedad de dispositivos como Smartphones, tabletas o portátiles integrarse eficientemente para compartir datos. Un ecosistema se construye haciendo que la infraestructura de negocio existente sea más móvil y le permita a los actores el acceso fácil de sus proyectos o archivos[47].

Actualmente existen tres ecosistemas principales: Apple, Google y Microsoft. A pesar de que son muy diferentes todos ellos dependen del almacenamiento en la nube, conocidos como Apple iCloud, Google Drive o Microsoft Skydrive. Los usuarios pueden almacenar los datos en un servidor remoto y los pueden acceder en cualquier lugar desde cualquier dispositivo.
Un ecosistema móvil se compone de[48]:

- Operadores de telefonía móvil: proporcionan servicios de acceso móvil a los usuarios.

- Redes de comunicación: los operadores proporcionan servicios a través de redes de telefonía celular de tercera 3G o cuarta 4G generación.

- Plataformas: son los framework de programación sobre los cuales se desarrollan las aplicaciones para los dispositivos móviles. Como todas las plataformas de software están divididas en tres categorías: licenciadas (Java ME, Windows Mobile y LiMo), propietarias (Blackberry, Iphone) y open source (Android).

- Sistemas operativos: ofrecen los servicios básicos y/o conjunto de herramientas que permiten a las aplicaciones comunicarse entre sí y compartir datos o servicios. Cada dispositivo trae instalado su propio sistema operativo, según la plataforma en la que están desarrollados.

- Aplicaciones: son las funcionalidades comunes que ofrece el sistema operativo o aquellas que pueden ejecutarse sobre el dispositivo móvil como juegos, navegador web, reproductor multimedia, etc.

En la figura 5 se puede apreciar un ecosistema móvil.

[47] THE NEXT WOMEN BUSINESS MAGAZIN. The Mobile Ecosystem: How Can it Benefit Your Business?. [en línea]. [citado el 17 de noviembre, 2014]. Disponible en internet: <http://www.thenextwomen.com/2013/01/09/mobile-ecosystem-how-can-it-benefit-your-business>
[48] NATIONAL TECHNOLOGY ASSISTANCE PROJECT. II. Understanding the Mobile Ecosystem. [en línea]. [citado el 17 de noviembre, 2014]. Disponible en internet: http://lsntap.org/book/export/html/3555

Figura 5. Ecosistema de un Smartphone

Fuente: Behance. Disponible en: https://www.behance.net/gallery/Smartphone-Ecosystem/6115789

Entre los principales beneficios de un ecosistema móvil podemos encontrar:

- La posibilidad de compartir información entre diferentes dispositivos de forma inalámbrica.
- Los datos en la nube permiten una fácil recuperación en caso de pérdida, daño o robo.
- Los datos de un dispositivo pueden ser visualizados en el mismo formato en otro dispositivo, lo cual es muy útil para realizar labores en movimiento. Esto ayuda a evitar la duplicidad de contenidos en las organizaciones.

En la figura 6, se visualiza el ecosistema Android y sus componentes.

Figura 6. Ecosistema Android

Fuente: Behance. Disponible en: https://www.behance.net/gallery/Smartphone-Ecosystem/6115789

Actualmente el ecosistema móvil Android presenta como principal desventaja la "fragmentación", entendida como la amplia variedad de dispositivos, versiones de sistema operativo, tamaños de pantalla, fabricantes, sensores, hardware entre otros.

La fragmentación es un fenómeno que ocurre cuando algunos usuarios móviles ejecutan versiones antiguas del sistema operativo y otros usuarios ejecutan versiones más recientes, con diferentes dispositivos de diferentes fabricantes.

La fragmentación se asocia generalmente con el sistema operativo Android porque los operadores inalámbricos y fabricantes de dispositivos son los que controlan cuando las actualizaciones del sistema son enviadas a los dispositivos y estos prefieren centrarse en la incorporación de nuevos clientes con la última versión de Android en vez de invertir tiempo y dinero en la construcción de actualizaciones del sistema operativo para los dispositivos más antiguos[49].

La fragmentación se convierte en un problema para las empresas de TI porque no pueden estandarizar una versión del sistema operativo y cada dispositivo tiene diferentes características; para los desarrolladores de aplicaciones porque deben crear diferentes versiones de la misma aplicación y asegurar que funciona correctamente en las diferentes versiones del sistema operativo, en las diferentes características de los dispositivos (hardware) y adaptarse a los diferentes tamaños de pantalla de los mismos.

Según informe de Open Signal[50] realizado en Agosto de 2014 existen más de 18.000 diferentes tipos de dispositivo con sistema operativo Android que pueden utilizar una misma aplicación haciendo que la optimización sea un reto grande. Comparado con el año 2013 la fragmentación aumento cerca del 60%.

4.1.11 Seguridad Informática. Disciplina informática que se encarga de diseñar las normas, procedimientos, métodos y técnicas destinados a conseguir un sistema de información seguro y confiable[51].

La seguridad informática tiene como muchos objetivos, entre ellos se destacan los siguientes[52]:

- Brindar mayores niveles de seguridad en la protección de los datos.

[49] TECHTARGET. Android fragmentation: More OS versions, more problems. [en línea]. [citado el 5 de Diciembre, 2014]. Disponible en: <http://searchconsumerization.techtarget.com/feature/Android-fragmentation-More-OS-versions-more-problems>
[50] OPEN SIGNAL. Android Fragmentation visualized. [en línea]. [citado el 27 de noviembre, 2014]. Disponible en internet: <http://opensignal.com/reports/2014/android-fragmentation/>
[51] AGUILERA, Purificación. Introducción a la Seguridad Informática. Editorial Editex. ISBN 978-84-9003-106-3. p. 9.
[52] GONZÁLEZ, Yina y CASTAÑO, Wilson. Fundamentos de Seguridad de la Información. Universidad Nacional Abierta y a Distancia. 2012. p. 37.

- Asegurar que la información que se transmite de un lugar a otro sea la misma que se envió y que realmente sea enviada y recibida por entes autorizados.
- Establecer los permisos de acceso a las aplicaciones informáticas existentes en una organización.
- Obligar a que los usuarios modifiquen sus claves en caso de que se haya realizado algún robo o conocimiento por otras personas ajenas a quien es responsable del manejo de la información.
- Ofrecer la confianza necesaria a los operadores y responsables de la información digital.
- Establecer los mecanismos de recuperación de información en caso de que haya ocurrido un siniestro.
- Garantizar la operación continua de los sistemas informáticos en cualquier organización.
- Definir políticas y normas de seguridad en las organizaciones.

Los Principios fundamentales de la seguridad informática son[53]:

- **Confidencialidad:** Cuando la información es solo accesible por aquellos a los cuales se ha autorizado a tener acceso.
- **Integridad:** Cuando la información es exacta y completa.
- **Disponibilidad:** Cuando la información es accedida solo por los usuarios que tienen los privilegios necesarios y suficientes para hacerlo.
- **Autenticación:** Cuando se puede garantizar la identidad de quien solicita acceso a la información.
- **No repudio:** Cuando la información involucrada en un evento corresponde a quien par cipa, quien no podrá evadir su intervención en éste.

4.1.11.1 Vulnerabilidad. El Open Web Application Security Project OWASP[54] define una vulnerabilidad como un “hueco” o debilidad de una aplicación, la cual puede ser un defecto de diseño o un bug (error) de la aplicación, que permite a un atacante causar daño a las partes que interactúan con la aplicación (desarrollador, usuarios y otras aplicaciones). Por ejemplo falta de validación de entrada de datos del usuario, falta de mecanismos de auditoria, manejo inadecuado de errores y cierre inadecuado de conexiones a base de datos.

4.1.11.2 Amenaza. Todo elemento o acción capaz de atentar contra la seguridad de la información. Las amenazas surgen a partir de la existencia de vulnerabilidades, es decir que una amenaza sólo puede existir si existe una

[53] Ibíd. p. 47.
[54] OWASP Open Web Application Security Project. Category: Vulnerability . [en línea]. [citado el 23 de octubre, 2014]. Disponible en internet: <https://www.owasp.org/index.php/Category:Vulnerability>

vulnerabilidad que pueda ser aprovechada, e independientemente de que se comprometa o no la seguridad de un sistema de información[55].

4.1.11.3 Riesgo. Probabilidad de que una amenaza se materialice, utilizando vulnerabilidad existentes de un activo o grupos de activos, generándoles pérdidas o daños[56].

Existen diferentes tipos de riesgos asociados a la seguridad de la información, entre los más comunes tenemos: riesgos de integridad (interface de usuario, procesamiento y administración), riesgos de relación, riesgos de acceso (procesos de negocio, aplicación, administración de la información, entorno de procesamiento, redes y nivel físico), riesgos de utilidad, riesgos en la infraestructura y riesgos de seguridad general.

4.1.11.4 Ataque. Consiste en aprovechar alguna debilidad o falla (vulnerabilidad) en el software, en el hardware, e incluso, en las personas que forman parte de un ambiente informático, a fin de obtener un beneficio, por lo general de índole económico, causando un efecto negativo en la seguridad del sistema, que luego repercute directamente en los activos de la organización[57].

Un ataque se compone por las siguientes fases:

- **Fase 1:** Reconocimiento. Obtención de información con respecto a una potencial víctima.

- **Fase 2:** Exploración. Se utiliza la información obtenida en la fase 1 para sondear el blanco y tratar de obtener información sobre el sistema víctima.

- **Fase 3:** Obtener acceso. Comienza a materializarse el ataque a través de la explotación de las vulnerabilidades y defectos del sistema descubiertos en las fases anteriores.

- **Fase 4:** Mantener el acceso. Una vez que el atacante ha conseguido acceder al sistema, buscará implantar herramientas que le permitan volver a acceder en el futuro desde cualquier lugar donde tenga acceso a Internet.

[55] UNIVERSIDAD NACIONAL DE LUJAN. Departamento de Seguridad Informática. Análisis a la seguridad de la información. [en línea]. [citado el 29 de octubre, 2014]. Disponible en internet: <http://www.seguridadinformatica.unlu.edu.ar/?q=node/12>

[56] CALDER, Alan y WATKINS, Steve. *IT Governance: A Manager's Guide to Data Security& BS 7799/ISO 17799*. Citado por SOLARTE, Francisco y BENAVIDES, Miriam. Riesgos y Control Informático. [en línea]. [citado el 29 de octubre, 2014]. Disponible en internet: <http://datateca.unad.edu.co/contenidos/233004/riesgos/leccin_3_anlisis_de_riesgos.html>

[57] MIERES, Jorge. Ataques informáticos. Debilidades de Seguridad comúnmente explotadas. [en línea]. [citado el 30 de noviembre, 2014]. Disponible en internet: <https://www.evilfingers.com/publications/white_AR/01_Ataques_informaticos.pdf>

- **Fase 5:** Borrar huellas. Una vez que el atacante logró obtener y mantener el acceso al sistema, intentará borrar todas las huellas que fue dejando durante la intrusión para evitar ser detectado por el profesional de seguridad o los administradores de la red.

Los tipos ataques informáticos se pueden clasificar en:

- Interrupción: cuando el recurso informático es destruido o no disponible.
- Interceptación: cuando la entidad no autorizada consigue acceso a un recurso (ataque contra la confidencialidad).
- Modificación: Cuando la entidad no autorizada consigue acceder a un recurso y es posible manipularlo (ataque contra la confidencialidad).
- Fabricación: Cuando la entidad no autorizada inserta objetos falsificados en el sistema (ataque contra la autenticidad).
- Monitorización: se realiza para observar a la víctima y su sistema, con el objetivo de establecer sus vulnerabilidades y posibles formas de acceso futuro.
- Autenticación: tiene como objetivo engañar al sistema de la víctima para ingresar al mismo. Generalmente este engaño se realiza tomando las sesiones ya establecidas por la víctima u obteniendo su nombre de usuario y password.

4.1.12 Seguridad en los dispositivos móviles. Con el auge de los dispositivos móviles, su uso generalizado y en continua expansión, con gran cantidad de información personal confidencial almacenada, en donde son usados para realizar todo tipo de transacciones online se han convertido en el principal blanco de los ciberdelincuentes.

Un aspecto importante en la seguridad en estos dispositivos son los canales de comunicación, ya que las amenazas pueden venir por: SMS, Bluetooth, Wi-Fi, navegadores, aplicaciones y correo electrónico, que puede propiciar la propagación de código malicioso orientado a este tipo de plataformas.

Según Himanshu Dwivedi, los principales problemas de seguridad que enfrentan los dispositivos móviles son[58]:

- La seguridad física de los dispositivos móviles debido al continuo incremento de la pérdida y robos.
- La seguridad en el almacenamiento del dispositivo.

[58] DWIVEDI, Himanshu. Mobile Aplication Security. Citado por RAMÍREZ, Gabriel. La seguridad en aplicaciones móviles: estrategias en el mundo actual. [en línea]. [citado el 18 de octubre, 2014]. Disponible en internet: <http://datateca.unad.edu.co/contenidos/233016/Articulos/La_Seguridad_en_Aplicaciones_Moviles_Estrategias_en_el_Mundo_Actual_Gabriel_Ramirez.pdf>. p. 3.

- Procesos de autenticación fuerte con contraseñas pobres.
- Soporte a múltiples usuarios con seguridad.
- Entornos de navegación seguros.
- Seguridad en sistemas operativos móviles.
- El aislamiento de las aplicaciones.
- La divulgación de información.
- Los Virus, Gusanos, Troyanos, Spyware y Malware.
- Los procesos de actualización y parcheo de los sistemas operativos.
- El uso y cumplimiento estricto del protocolo SSL.
- Phishing.
- Solicitud de falsificación de sitio cruzado.
- La localización privacidad y seguridad.
- Drivers de dispositivos inseguros.
- Múltiples factores de autenticación.

4.1.13 Tipos de Ataques a dispositivos móviles. Los ataques a los dispositivos móviles están asociados a diferentes riesgos y vulnerabilidades que se pueden presentar con el uso de los dispositivos, en este sentido los tipos de ataques se pueden agrupar de diferentes formas y dependiendo del enfoque que se le desee dar a los ataques de los dispositivos móviles.

Los puntos de ataque para los dispositivos móviles son[59]:

- Las credenciales y los servicios externos del dispositivo como el correo electrónico, las cuentas de bancos, etc.
- Los datos personales de los usuarios.
- Los datos de los dispositivos como los números de cuenta, números de las tarjetas y las fechas de expiración.
- Acceso al dispositivo para revisar la simcard del dispositivo, revisión de las conexiones telefónicas y de internet, uso del dispositivo para enviar virus, malware y procesamiento de actividades, robo de datos secretos y datos sensibles del dispositivo.
- Almacenamiento de datos, robo, revisión y modificación de claves, información de las bases de datos, archivos de configuración, archivos de las aplicaciones, las caches de los sistemas.
- Archivos binarios, realización de ingeniería inversa para entender el binario, búsqueda de las vulnerabilidades que pueden ser explotadas, incrustar credenciales y generación automática de claves.
- Plataformas móviles de enganche de las plataformas, instalación de malware, aplicaciones móviles de ejecuciones automáticas no autorizadas, las decisiones de la arquitectura de aplicaciones basadas en la plataforma.

[59] RAMÍREZ. Op. cit., p. 126.

- El almacenamiento de datos, los archivos binarios y la plataforma no son independientes y se encuentran relacionados entre sí.

Entre los tipos de ataques que se pueden presentar a dispositivos móviles se pueden clasificar en[60]:

- Vulnerabilidades del navegador: estos exploits están diseñados para aprovechar vulnerabilidades del software utilizado para acceder a sitios web. Visitar ciertas páginas web y/o hacer clic en hipervínculos puede desencadenar ciertas vulnerabilidades del navegador que permiten la instalación de malware o realizar otras acciones desfavorables en el dispositivo móvil.

- Interceptación de datos: puede ocurrir cuando un atacante realiza eavesdropping a las comunicaciones procedentes desde el dispositivo móvil. Esto es posible a través de diversas técnicas, como man-in-the-middle, Wi-Fi sniffing, etc.

- Keylogger: es un tipo de malware que registra las pulsaciones de teclado en los dispositivos móviles para capturar información confidencial, como números de tarjetas de crédito o contraseñas.

- Malware: software malicioso que se disfraza como un juego, parche, utilidad u otra aplicación de terceros para iniciar una variedad de ataques y extenderse a otros dispositivos. Entre las versiones de malware se puede encontrar virus, troyanos, gusanos, spyware, etc.

- Seguimiento de ubicación no autorizado: permite conocer y monitorear la ubicación del dispositivo móvil registrado.

- Explotación de la red: este ataque tiene en cuenta las vulnerabilidades del sistema local (por ejemplo bluetooth) o redes de telefonía móvil. A menudo puede tener éxito sin la intervención del usuario, por lo que es peligroso cuando se usa para propagar malware. Con herramientas especiales, los atacantes pueden encontrar los usuarios en una red wi-fi, secuestrar credenciales de usuario y usarlas para la suplantación de identidad.

- Phishing: es una estafa que con frecuencia utiliza el correo electrónico o mensajes emergentes para engañar a las personas para que revelen información confidencial.

[60] TEKADE Puja S. & SHELKE C.J. A Survey on different Attacks on Mobile Devices and its Security. [en línea] [citado el 8 octubre, 2014]. Disponible en internet: < http://www.ijaiem.org/volume3issue2/IJAIEM-2014-02-28-080.pdf>

- Spamming: es publicidad comercial de productos, servicios y sitios web recibida por correo electrónico y/o mensajes de texto. También puede ser usado como mecanismo de entrega de software malicioso.

- Spoofing: los atacantes pueden crear sitios web fraudulentos de sitios legítimos para engañar a los usuarios y distribuir malware a los dispositivos móviles.

- Robo/pérdida: debido a su pequeño tamaño y su uso fuera de la oficina, los dispositivos móviles puede ser más fáciles de extraviar o robar que un computador. Si los dispositivos móviles se pierden o son robados, puede ser relativamente fácil obtener acceso a la información que almacenan.

- Exploit de día cero: se aprovecha de una vulnerabilidad de seguridad antes que haya una actualización disponible.

4.1.14 Anatomía de un ataque móvil. Un ataque móvil contiene tres puntos sensibles que los delincuentes buscan explotar: el dispositivo, la red y el centro de datos, o una combinación de éstos[61] [62]. En la figura 7 se ilustra un ataque móvil.

Figura 7. Anatomía de un ataque móvil

Fuente Viaforensics. Disponible en: https://www.nowsecure.com/resources/downloads/secure-mobile-development/

[61] VIAFORENSICS. 42+ Best practices: Secure mobile development for iOS and Android. 2012. [en línea] [citado el 8 octubre, 2014]. Disponible en internet: <https://viaforensics.com/resources/reports/best-practices-ios-android-secure-mobile-development/mobile-security-primer/>

[62] NOWSECURE. Secure Mobile Development Best Practices. [en línea] [citado el 9 diciembre, 2014]. Disponible en internet: <https://www.nowsecure.com/resources/downloads/secure-mobile-development/>

Los ataques centrados en el *dispositivo* usan varios puntos de entrada como el navegador, correo electrónico, el teléfono, los mensajes SMS, las aplicaciones de terceros, el sistema operativo, banda base de radiofrecuencia y otros canales de comunicación como bluetooth, encontrándose entre los más comunes: phishing, framing, Clickjacking, Drive-by Downloading, Main in the mobile (MitMo), ataques en la banda base (GSM, 3G), SMiShing, ataques por Radio Frecuencia (Bluejacking, NFC), almacenamiento de datos sensibles, aplicaciones sin cifrado o con cifrado débil, validación SSL inapropiada, manipulación de la configuración, Inyección de código en tiempo de ejecución, mala configuración de permisos en aplicaciones, aumento de privilegios, no uso de patrón o código de acceso al dispositivo, jailbreak, rooteo del dispositivo, contraseñas y datos accesibles, exploits del día cero[63].

Los ataques a la *red* tienen como puntos de entrada wi-fi sin encriptación, puntos de acceso no autorizados, escaneo de paquetes, Man in the Middle (MITM), SSLStrip, secuestro de sesión, envenenamiento DNS, Certificados SSL falsos, masquereading, Denegación de Servicio DoS, Eavesdropping y confidencialidad de posicionamiento[64].

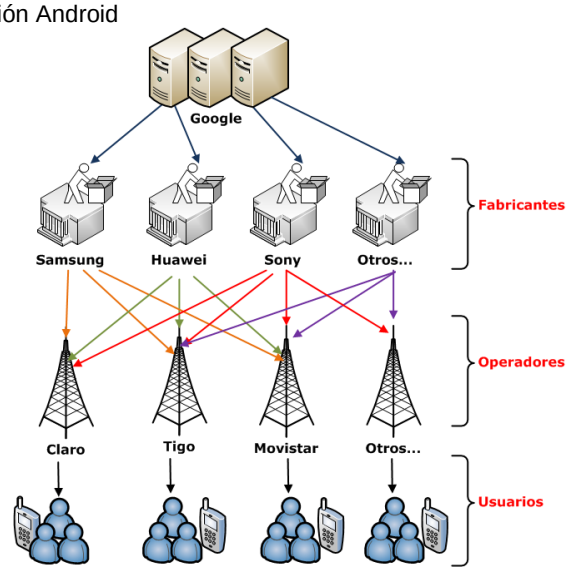
Los ataques orientados al *centro de datos* tienen como puntos principales de entrada el servidor Web y la Base de Datos encontrándose vulnerabilidades en la plataforma, en la configuración del servidor, Cross-site scripting (XSS), Cross-site Request Forgery (CSRF), validación débiles de entradas de usuario, ataques de fuerza bruta, inyección de código SQL, ejecución de comandos del sistema operativo, escalada de privilegios y volcado de datos[65].

4.1.15 Riesgos de seguridad a nivel de sistema operativo. Los defectos en el sistema operativo son muy comunes y actualmente son el blanco de los atacantes que desean causar un alto impacto en el sistema.

Los sistemas operativos se actualizan periódicamente con mejoras en funcionalidad, parches y correcciones de seguridad los cuales pueden o no coincidir con las modificaciones realizadas en el firmware del fabricante del dispositivo móvil generando un riesgo de seguridad. Los fabricantes de dispositivos u operadores de telefonía móvil modifican el sistema operativo para incluir información o funciones propias y estos cambios pueden generar huecos de seguridad debido a que el sistema operativo se modifica sin saber lo que sucederá en el dispositivo. En la figura 8 se muestra el modelo de actualización del sistema operativo Android donde intervienen los elementos del ecosistema móvil.

[63] Ibíd.
[64] Ibíd.
[65] Ibíd.

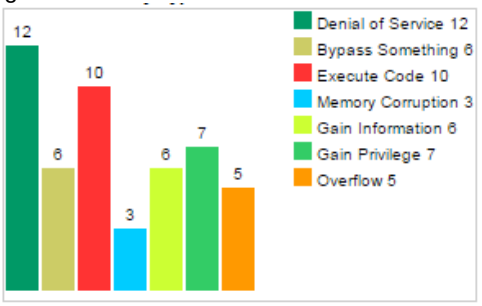
Figura 8. Modelo Actualización Android



Fuente: Los Autores

Entre las vulnerabilidades más significativas del sistema operativo Android durante el último Año se encuentran: Denegación del servicio, salto de controles (bypass something), ejecución de código, corrupción de memoria, obtención de información, obtención de privilegios y overflow como se muestra en la figura 9[66].

Figura 9. Vulnerabilidades más significativas en Android



Fuente: CVE. Disponible en: http://www.cvedetails.com/product/19997/Google-Android.html?vendor_id=1224

4.1.16 Riesgos potenciales de las aplicaciones móviles. Las aplicaciones móviles pueden acceder a sistemas de servidores, almacenamiento y redes vitales para la seguridad. Un atacante capaz de explotar una aplicación puede acceder a estos sistemas o invalidarlos. Además de atacar un sistema, desconfigurar una página web y robar datos de ésta, las aplicaciones móviles son capaces de acceder a libretas de direcciones, descubrir datos de ubicación, enviar mensajes

[66] CVE DETAILS. Android: Vulnerability Statistics. [en línea] [citado el 8 diciembre, 2014]. Disponible en internet: <http://www.cvedetails.com/product/19997/Google-Android.html?vendor_id=1224>

de texto, hacer llamadas y acceder a las redes internas. Cada tipo de aplicación móvil presenta un conjunto de riesgos ligeramente distinto porque cada uno tiene un diseño y capacidades diferentes[67].

4.1.16.1 Riesgos de seguridad en aplicaciones web[68]. Las aplicaciones web tienen dos componentes principales: el servidor y el cliente. Las vulnerabilidades del servidor pueden estar presentes en la parte de la aplicación que se ejecuta en el servidor. Las vulnerabilidades del cliente pueden ser explotadas potencialmente dentro de la página web cuando ésta es representada y ejecutada dentro de un navegador web.

En el lado del servidor, éste puede aceptar datos de clientes que no son de confianza y procesar dichos datos para devolver una respuesta al cliente. Estos datos no verificados pueden emplearse para acceder a una base de datos, un sistema de archivos u otras fuentes de información vital para la seguridad. Si el servidor no limpia adecuadamente los datos no verificados, éstos podrían provocar el deterioro de la base de datos, exponer archivos confidenciales o abrir la puerta a daños de otra clase[69].

En el lado del cliente, ejecutar una página web enviada desde el servidor generalmente implica cargar la página y ejecutar JavaScript. El cliente ejecuta todo el código en el contexto de un origen específico, por lo que si se ejecutan de algún modo datos no verificados de un atacante, el atacante goza de plena potestad para acceder a la página y modificarla. Esto significa que los atacantes pueden capturar pulsaciones de teclas, robar los datos introducidos, alterar la página o ejecutar un ataque de *phishing* convincente[70].

4.1.16.2 Riesgos de seguridad en aplicaciones nativas[71]. Las aplicaciones nativas tienen sus propias preocupaciones en cuanto a la seguridad, que generalmente pueden clasificarse en dos categorías: riesgos para la aplicación y riesgos para el dispositivo móvil. Un riesgo para la aplicación es todo aquel que pueda poner en peligro información confidencial o a la aplicación misma. Un riesgo para el dispositivo móvil es todo aquel que puede tener lugar fuera de la aplicación, como enviar mensajes de texto, agotar la batería o realizar llamadas telefónicas.

[67] IBM Corporation. Op. cit. p. 5.
[68] Ibíd.
[69] Ibíd.
[70] Ibíd.
[71] Ibíd.

Una vez los atacantes tengan acceso a una aplicación que puedan explotar, podrán abusar de ésta hasta que el usuario les detenga activamente o bien la aplicación sea corregida por los desarrolladores y actualizada por el usuario.

4.1.16.3 Riesgos de seguridad en aplicaciones híbridas[72]**.** Debido a que las aplicaciones híbridas son en parte aplicación nativa y en parte aplicación web, reúnen los riesgos para la seguridad de ambos tipos de aplicación.

Las aplicaciones híbridas basadas en HTML5 son más susceptibles a cualquier tipo de ataque que las aplicaciones nativas, lo que puede resultar en la captura de información personal y el posterior envío a través de un software malicioso a los contactos de las víctimas en un mensaje de texto.

A diferencia de las aplicaciones nativas que sólo muestran el posible código malicioso, la aplicación basada en HTML5, dependiendo del JavaScript API, ejecuta directamente el código.

4.1.17 Proyecto Seguridad Móvil OWASP (Mobile Security Project). OWASP (Open Web Application Security Project) es un proyecto abierto dedicado a buscar y combatir las causas de inseguridad en el desarrollo de software, proporcionando gran cantidad de documentación y herramientas a los desarrolladores y equipos de seguridad para construir y mantener aplicaciones seguras[73].

OWASP contiene un proyecto de Seguridad móvil con el objetivo de clasificar los riesgos de seguridad móvil y proporcionar controles de desarrollo para reducir su impacto o la probabilidad de explotación[74].

El enfoque principal está en la capa de aplicación. No solo se centra en las aplicaciones móviles desarrolladas para el usuario final, sino también en el lado del servidor y la infraestructura con que las aplicaciones móviles se comunican. En gran medida se centra en la integración entre la aplicación móvil, servicios de autenticación remota y las características específicas de la plataforma en la nube[75].

En la figura 10 se presenta el Top 10 de riesgos de seguridad móvil del año 2014 identificados por OWASP, de acuerdo a encuestas de las nuevas estadísticas de vulnerabilidad en el campo de las aplicaciones móviles.

[72] Ibíd.
[73] OWASP. OWASP Mobile Security Project. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: <https://www.owasp.org/index.php/OWASP_Mobile_Security_Project/>
[74] Ibíd.
[75] Ibíd.

Figura 10. Top 10 Riesgos de seguridad móvil identificados por OWASP

Fuente: Projects/OWASP Mobile Security Project - Top Ten Mobile Risks. Disponible en internet: https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Top_Ten_Mobile_Risks

A continuación, se identifican los diez riesgos más importantes para las aplicaciones móviles[76]:

1. M1 Debilidad en los controles del lado del servidor de la aplicación (Weak Server Side Controls): corresponde a las inyecciones de código. Los atacantes buscan vulnerabilidades en los servidores para poder explotarlas inyectando códigos maliciosos.

2. M2 Almacenamiento de datos inseguro (Insecure Data Storage): se trata de dispositivos móviles perdidos y/o robados, aunque también está la posibilidad de acceder a dichos dispositivos sin la necesidad de tenerlos físicamente a través de *exploits in-the wild* y/o distintos códigos maliciosos.

3. M3 Protección insuficiente en la capa de transporte (Insufficient Transport Layer Protection): cuando se desarrolla una aplicación normalmente los datos son intercambiados entre un cliente y un servidor. Si la codificación de dicha aplicación es débil, existen diversas técnicas para visualizar datos sensibles mientras viajan entre el cliente y el servidor.

4. M4 Fuga de datos involuntaria (Unintended Data Leakage): las aplicaciones móviles tienen que interactuar con sistemas operativos, infraestructuras digitales, *hardwares* nuevos, etc., que no son propiedad de los desarrolladores, por lo que no pueden controlar cambios y/o fallas que estén

[76] ESET Latinoamerica. Top 10 de OWASP de vulnerabilidades en aplicaciones móviles. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: <http://www.welivesecurity.com/la-es/2014/02/26/top-10-owasp-vulnerabilidades-aplicaciones-moviles/>

por fuera de sus aplicaciones. En este sentido, es posible que se pierdan datos si no se realizan evaluaciones para entender cómo las aplicaciones interactúan con todos los elementos de los dispositivos.

5. M5 Autenticación y autorización pobres (Poor Authorization and Authentication): existen patrones de autenticación considerados inseguros y que deben ser evitados. Algunos ejemplos son: “Recuérdame” (cuando existe la opción de que la aplicación guarde la contraseña de forma automática), la falta de *tokens* de seguridad**,** etc.

6. M6 Criptografia rota (Broken Cryptography): en algunas ocasiones, los métodos de encriptación de datos se vuelve una práctica casi obsoleta. Crear y utilizar su propio algoritmo de encriptación y utilizar algoritmos desfasados son ejemplos de malas prácticas.

7. M7 Inyección del lado del cliente (Client Side Injection): siempre y cuando exista la posibilidad de que usuarios externos, internos y la aplicación misma puedan enviar datos no confiables al sistema, un atacante podría inyectar *exploits* sencillos a las aplicaciones móviles, lo que causa un potencial riesgo de robo de información.

8. M8 Decisiones de seguridad vía entradas no confiables (Security Decisions Via Untrusted Inputs): Los procesos entre aplicaciones y sistemas operativos comparten espacios de memoria para permitir la comunicación y sincronización entre los mismos. Para minimizar los riesgos de ataque, la aplicación móvil debería permitir solamente comunicación con otras aplicaciones confiables, las acciones sensibles deberían requerir la interacción del usuario, la información sensible no debería ser enviada a través de IPC (comunicación entre procesos), etc.

9. M9 Manejo de sesiones inapropiado (Improper Session Handling): el manejo incorrecto de la información es muy similar a la autenticación débil (M5). Es tan importante manejar bien la sesión una vez abierta como establecer la misma sesión. Si no se aplican cuidados sencillos pero importantes como validar la sesión a nivel servidor y no solamente a nivel cliente, establecer un tiempo de expiración de sesión o la creación de *tokens* seguros puede que terceros no autorizados accedan a información de otros usuarios.

10. M10 Falta de protección de los binarios (Lack of Binary Protections): la falta de protección a nivel binario facilita el ataque a través de ingeniería reversa. Si un programador no es creador del código de su programa a nivel binario y no lo tiene protegido, un atacante puede fácilmente buscar fallas en el código, copiarlo, hacer cambios menores y revender una aplicación móvil nueva como si fuese suya.

A continuación se detallan los riesgos M2 y M3, los cuales serán evaluados en el presente estudio teniendo en cuenta que su explotabilidad representa un alto impacto para el usuario y la información que se maneja en el dispositivo.

4.1.17.1 M2 Almacenamiento de datos inseguro (Insecure Data Storage).[77] Ocurre cuando el desarrollador asume que los usuarios, otras aplicaciones o malware no tendrán acceso al sistema de archivos del dispositivo móvil y a la información sensible almacenada en el mismo. Los sistemas de archivos son fácilmente accesibles. El rooteo de un dispositivo elude cualquier protección de cifrado, y cuando los datos no están protegidos adecuadamente, con herramientas especializadas es posible ver los datos de la aplicación.

Los datos almacenados de forma insegura a menudo son:

- Nombres de usuario.
- Tokens de autenticación.
- Contraseñas.
- Cookies.
- Datos de ubicación.
- Identificador del dispositivo, nombre del dispositivo, nombre de la conexión de red.
- Información personal: dirección, datos tarjeta de crédito, etc.
- Bases de datos SQLite.
- Datos de la aplicación: archivos Log, archivos Plist, archivos manifest o xml, almacenes de datos binarios, historial de transacciones, información de depuración, etc.
- SD card.
- Sincronización en la nube.

Un almacenamiento inadecuado de los datos se considera un factor crítico en la seguridad informática teniendo en cuenta los conceptos de Confidencialidad e Integridad de la misma, razón por la cual en este estudio se analizará el comportamiento de las aplicaciones respecto al manejo y administración de los datos sensibles, revisando si la información es almacenada de forma cifrada o si por el contrario está expuesta de forma insegura en el dispositivo, con el fin de determinar las vulnerabilidades que puedan afectar el sistema operativo y al usuario.

[77] OWASP. Mobile Top 10 2014-M2 Insecure Data Storage. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: <https://www.owasp.org/index.php/Mobile_Top_10_2014-M2>

4.1.17.2 M3 Protección insuficiente en la capa de transporte (Insufficient Transport Layer Protection).[78] Las aplicaciones con frecuencia no protegen el tráfico de la red. Pueden utilizar SSL/TLS durante la autenticación, pero no en otros puntos del flujo de la misma. Estas prácticas de desarrollo conducen al riesgo de interceptación de datos y la información de sesiones. Usar seguridad en la capa de transporte no significa que en la aplicación se ha implementado correctamente.

Para detectar fallas básicas y observar el tráfico de red de un dispositivo móvil, se requiere inspeccionar el diseño y configuración de la aplicación.

Este defecto expone los datos de un usuario individual y puede conducir al robo de cuenta. Si el delincuente intercepta una cuenta de administrador, todo el sitio podría estar expuesto. La mala configuración SSL también puede facilitar los ataques de phishing y MITM (Man in the mobile)

Las aplicaciones móviles envían información sensible (contraseñas, cuentas de usuario, datos de tarjetas de crédito, etc.) a través de la red mediante el uso de protocolos (seguros o inseguros), pero no es posible garantizar que las aplicaciones usen adecuadamente estos protocolos, poniendo en riesgo la confidencialidad de la información, razón por la cual se analizará el comportamiento de las aplicaciones al transportar información con el fin de identificar las vulnerabilidades que se puedan presentar.

4.1.18 Sistema Operativo Android. Como se ha comentado, existen muchas plataformas para Smartphone; sin embargo, Android presenta una serie de características que lo hacen diferente. Es el primero que combina en una misma solución las siguientes cualidades[79]:

- El código fuente de Android es abierto, lo que significa que cualquier interesado puede descargar y construir su propio sistema Android. Sin embargo, no todas las partes de Android están abiertas, las aplicaciones de google que vienen instaladas por defecto en la mayoría de dispositivos móviles Android, son modificadas por el fabricante del dispositivo para adaptarse mejor a su hardware, incluyendo controladores propietarios de código cerrado.

- Es una plataforma de desarrollo libre basada en Linux, que proporciona una forma para que las aplicaciones interactúen con el hardware, así como la gestión de procesos y memoria. Las versiones de Android

[78] OWASP. Mobile Top 10 2014-M3 Insuficient Transport Layer. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: <https://www.owasp.org/index.php/Mobile_Top_10_2014-M3>
[79] GIRONES, Jesús. El gran libro de Android. Ediciones Marcombo. Tercera Edición. 2013. ISBN: 978-84-267-2078-8.

anteriores al 4.0 usan la versión del kernel 2.6 y las posteriores usan el kernel 3.x.

- El modelo de seguridad de Android es basado en permisos[80]. Esto significa que para que una aplicación realice una acción se debe otorgar explícitamente el permiso para llevarla a cabo. Esto significa que, al igual que en un sistema UNIX, el sistema operativo Android requiere que cada aplicación se ejecute con su propio identificador de usuario (UID) e identificador de grupo (GID).Estos permisos se aplican en la arquitectura Android a nivel de núcleo y a nivel del framework de aplicaciones.

- Adaptable a cualquier tipo de hardware. Android no ha sido diseñado exclusivamente para su uso en teléfonos y tabletas. Este hecho tiene sus evidentes ventajas, pero también va a suponer un esfuerzo adicional al programador. La aplicación ha de funcionar correctamente en dispositivos con gran variedad de tipos de entrada, pantalla, memoria, etc.

- Portabilidad asegurada. Las aplicaciones finales son desarrolladas en Java lo que asegura que podrán ser ejecutadas en cualquier tipo de CPU, tanto presente como futuro. Esto se consigue gracias al concepto de máquina virtual.

- Arquitectura basada en componentes inspirados en Internet. Por ejemplo, el diseño de la interfaz de usuario se hace en xml, lo que permite que una misma aplicación se ejecute en un móvil de pantalla reducida o en un TV.

- Filosofía de dispositivo siempre conectado a Internet.

- Gran cantidad de servicios incorporados. por ejemplo, localización basada tanto en GPS como en redes, bases de datos con SQL, reconocimiento y síntesis de voz, navegador, multimedia.

- Aceptable nivel de seguridad. Los programas se encuentran aislados unos de otros gracias al concepto de ejecución dentro de una caja que hereda de Linux. Además, cada aplicación dispone de una serie de permisos que limitan su rango de actuación (servicios de localización, acceso a Internet, etc.).

- Optimizado para baja potencia y poca memoria. Por ejemplo, Android utiliza la Máquina Virtual Dalvik. Se trata de una implementación de

[80] BERGMAN, Neil et al. Hacking Exposed Mobile. Security secrets & solutions. Editorial Mc Graw Hill. 2013. ISBN: 978-0-07-181702-8

Google de la máquina virtual de Java optimizada para dispositivos móviles.

- Alta calidad de gráficos y sonido. gráficos vectoriales suavizados, animaciones inspiradas en Flash, gráficos en 3 dimensiones basados en OpenGL. Incorpora codecs estándar más comunes de audio y vídeo, incluyendo H.264 (AVC), MP3, AAC, etc.

4.1.18.1 Arquitectura Sistema Operativo Android. Android es una plataforma para dispositivos móviles que contiene un software en estructura de pila que incluye un sistema operativo, software para conectar aplicaciones (middleware) y aplicaciones base.

La arquitectura Android está formada por capas[81]:

- Aplicaciones: Este nivel contiene, tanto las incluidas por defecto de Android como aquellas que el usuario vaya añadiendo posteriormente, ya sean de terceras empresas o de su propio desarrollo. Todas estas aplicaciones utilizan los servicios, las API y librerías de los niveles anteriores.

- Framework de Aplicaciones: Representa fundamentalmente el conjunto de herramientas de desarrollo de cualquier aplicación. Toda aplicación que se desarrolle para Android, ya sean las propias del dispositivo, las desarrolladas por Google o terceras compañías, o incluso las que el propio usuario cree, utilizan el mismo conjunto de API y el mismo "framework", representado por este nivel.

- Librerías nativas: incluye un conjunto de librerías escritas utilizando C/C++ y proporcionan a Android la mayor parte de sus capacidades más características. Junto al núcleo basado en Linux, estas librerías constituyen el corazón de Android. Los desarrolladores utilizan estas bibliotecas a través del framework de aplicaciones de Android.

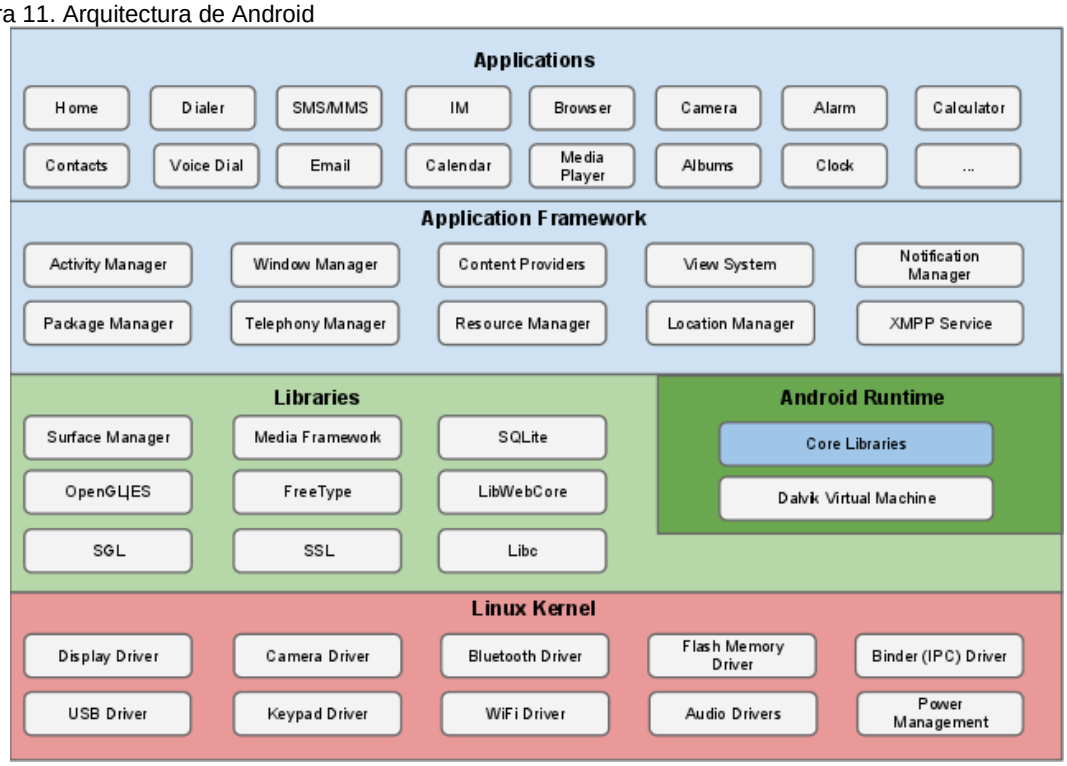
- Runtime de Android: Al mismo nivel que las librerías de Android se sitúa el entorno de ejecución. Éste lo constituyen las Core Libraries, que son librerías con multitud de clases Java y la máquina virtual Dalvik. Algunas librerías son System C library, Media Framework, Surface Manager, WebKit, SGL, Librerías 3D, FreeType, SQLite, SSL

[81] UNIVERSIDAD CARLOS III DE MADRID. Programación en dispositivos móviles portables. Arquitectura Android [en línea]. [citado el 28 Mayo, 2014]. Disponible en internet: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android

- Núcleo Linux: Android utiliza el núcleo de Linux como una capa de abstracción para el hardware disponible en los dispositivos móviles. Esta capa contiene los drivers necesarios para que cualquier componente hardware pueda ser utilizado mediante las llamadas correspondientes. Siempre que un fabricante incluye un nuevo elemento de hardware, lo primero que se debe realizar para que pueda ser utilizado desde Android es crear las librerías de control o drivers necesarios dentro de este kernel de Linux embebido en el propio Android.

Cada una de estas capas utiliza servicios ofrecidos por las anteriores, y ofrece a su vez características a las capas de niveles superiores, como se muestra en la figura 11.

Figura 11. Arquitectura de Android



Fuente: Android. Android Security Overviews. [en línea] Disponible en internet: https://source.android.com/devices/tech/security/index.html

4.1.18.2 Modelo de Seguridad del Sistema Operativo Android. El principio fundamental de seguridad de Android[82] es que una aplicación maliciosa no afecte los recursos del sistema operativo, al usuario y a otras aplicaciones. Para mantener este principio, siendo Android un sistema operativo multicapa, se aplican

[82] ZHAUNIAROVICH, Yury. AndroidTM Security (and Not) Internals. 2014. License Creative Commons. p. 7.

mecanismos de seguridad en todos los niveles lo cual proporciona flexibilidad y brinda protección a los datos del usuario y a los recursos del sistema.

Las principales características de seguridad de Android son:

- *Sandbox* obligatorio para todas las aplicaciones.
- Separación de privilegios para asegurarse que ninguna aplicación pueda leer o escribir código o datos de otras aplicaciones, el usuario del dispositivo o el sistema operativo en sí mismo.
- Control de acceso basado en usuarios y grupos. Los recursos y operaciones proporcionadas por el kernel son restringidos basados en los permisos que se han concedido al usuario.
- En Android todas las aplicaciones se les asigna un identificador de usuario único, restringiéndolas a que solo puedan acceder a los recursos y funcionalidades concedidas.
- En el framework de aplicaciones, una aplicación para Android debe declarar un permiso en su archivo de manifiesto (AndroidManifest.xml). Estos permisos son solicitados al usuario durante la instalación de la aplicación, dándole la opción de instalarla con los permisos solicitados o la no instalación de la misma.

A nivel del Kernel de Linux, cada aplicación se ejecuta en un Sandbox de aplicación especial; el Kernel fuerza el aislamiento de las aplicaciones y los componentes del sistema operativo haciendo uso de funcionalidades básicas de Linux como separación de procesos y Control de Acceso Discrecional (ACL). El aislamiento se impone asignando a cada aplicación un identificador único de usuario (UID), cuando una aplicación es instalada Android asigna un nuevo UID, asegurándose que dicho identificador no está siendo usado en el dispositivo, y la nueva aplicación se ejecutara bajo ese UID, así mismo, a todos los datos almacenados por la aplicación se les asignara el mismo UID[83].

Gracias a este mecanismo se logra[84]:

- Que las aplicaciones no interfieran entre ellas y tengan acceso limitado a los recursos que ofrece el sistema operativo.
- Que una aplicación A no acceda a los archivos de una aplicación B.
- Que una aplicación A no consuma espacios de memoria que use una aplicación B.
- Que una aplicación A no acceda al mismo tiempo a las mismas características que use una aplicación B (ejemplo cámara, GPS, bluetooth).

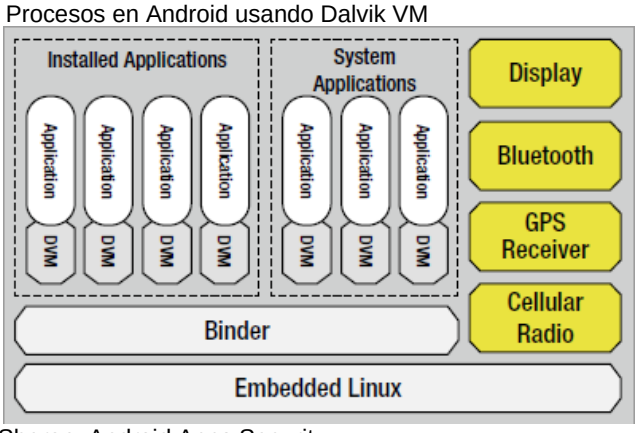
[83] SIX, Jeff. Application.Security for the Android Platform. Editores O'Reilly. 2012. p. 15. ISBN. 978-1-449-31507-8

[84] ANDROID. Android Security Overview. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: <http://source.android.com/devices/tech/security/index.html

Las aplicaciones de Android se ejecutan, de forma predeterminada, en un sandbox con pocos privilegios. Por lo tanto, una aplicación sólo tiene acceso a un limitado conjunto de las características del sistema. El sistema operativo Android controla el acceso de las aplicaciones a los recursos del sistema que pueden afectar negativamente la experiencia del usuario, también controla el acceso a las características como GPS, Cámara, o marcación telefónica, las cuales pueden ser accedidas por aplicaciones de terceros.

Todas las aplicaciones de Android se ejecutan en la máquina virtual Dalvik (DVM). El DVM es donde el código en bytes, o los bloques fundamentales de código, se ejecutarán. Es análoga a la Máquina Virtual Java (JVM) que existe en los computadores personales y servidores hoy en día. En la figura 12 se muestra como cada aplicación se ejecuta en una instancia propia de la máquina Dalvik[85].

Figura 12. Separación de Procesos en Android usando Dalvik VM

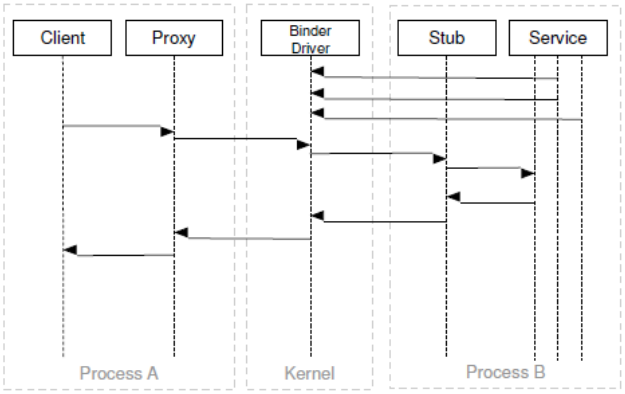


Fuente: GUNASEKERA, Sheran. Android Apps Security

Con la separación de procesos, se requiere un mecanismo para organizar la comunicación y el intercambio de datos y mensajes entre los diferentes procesos, para lograr esto, Android implementa IPC (Comunicación Inter Procesos) mediante un framework llamado *Binder*, el cual ofrece las funciones necesarias para organizar todo tipo de comunicación entre procesos en el sistema operativo; *Binder* proporciona características como la posibilidad de invocar métodos de objetos remotos como si fueran locales, invocación de métodos sincrónica y asincrónica, capacidad para enviar descriptores de archivo a través de procesos, etc[86]. En la figura 13 se muestra el modelo de comunicación Binder.

[85] GUNASEKERA, Sheran. Android Apps Security. Editorial Apress. 2012. p. 32. ISBN 978-1-4302-4063-1.
[86] ZHAUNIAROVICH. Op. cit., p. 30.

Figura 13. Android Binder communication model



Fuente: ZHAUNIAROVICH, Yury. AndroidTM Security (and Not) Internals

Las aplicaciones necesitan aprobación para llevar a cabo las tareas que el usuario requiere, como el envío de mensajes SMS, utilizar la cámara, o el acceso a la base de datos de contactos. Android utiliza un manifiesto de permisos para realizar seguimiento a lo que el usuario permite que hagan las aplicaciones. Los permisos que una aplicación requiere se expresan en el archivo *AndroidManifest.xml*, y el usuario los acepta al instalar la aplicación.[87]

Existen cuatro niveles de protección para los permisos[88]:

- *Normal*: es el valor por defecto. Es un permiso de menor riesgo que permite a las aplicaciones solicitar características del nivel de aplicación, con un riesgo mínimo para otras aplicaciones, el sistema o el usuario. El sistema otorga, de forma automática este tipo de permiso en la instalación de una aplicación, sin pedir la aprobación explícita del usuario (aunque el usuario siempre tiene la opción de revisar estos permisos antes de instalar).

- *Peligrosa* (Dangerous): Un permiso de alto riesgo que le daría acceso a las aplicaciones a solicitar los datos privados del usuario o el control sobre el dispositivo.

- *Firma* (Signature): Un permiso que el sistema otorga sólo si la aplicación que solicita está firmada con el mismo certificado de la aplicación que declaró el permiso. Si los certificados coinciden, el sistema otorga automáticamente el permiso sin notificar al usuario o pidiendo la aprobación explícita del usuario.

[87] HIMANSHU, Dwivedi. Mobile Application Security. Editorial Mc Graw Hill. P. 22 ISBN 978-0-07-163356-7.
[88] ANDROID INC. <Permission>. [en línea] [citado el 16 mayo, 2014]. Disponible en internet: <https://developer.android.com/guide/topics/manifest/permission-element.html#plevel>

- *Firma o Sistema* (SignatureOrSystem): Un permiso que el sistema otorga sólo a las aplicaciones que se encuentran en la imagen del sistema Android o que se firman con el mismo certificado de la aplicación que declaró el permiso.

La arquitectura y el modelo de seguridad del sistema operativo Android ha sido diseñada, estructurada y desarrollada buscando aumentar los niveles de protección y seguridad del sistema operativo y la información gestionada; la segmentación por capas, la solicitud de permisos para las aplicaciones, la ejecución en sandbox independientes y el control de acceso de usuarios constituyen la base de seguridad del sistema operativo, en este punto la capa de aplicaciones representa un mayor riesgo debido a que es donde se genera una constante interacción con otros actores (fabricantes, proveedores y desarrolladores) que pueden omitir la implementación de medidas de seguridad en las aplicaciones liberadas aumentando las vulnerabilidades que pueden ser explotadas por los ciberdelincuentes para acceder a la información de la aplicación misma o para obtener acceso a otras capas del sistema operativo.

4.1.19 Problemas de seguridad en Java. Java ofrece un framework eficiente para desarrollar y desplegar aplicaciones empresariales, de servidor o de cliente. Java es la base para prácticamente todos los tipos de aplicaciones de red, además del estándar global para desarrollar y distribuir aplicaciones móviles y embebidas, juegos, contenido basado en web y software de empresa. Sin embargo, el código de bytes de Java es un lenguaje interpretado, contiene metadatos muy detallados y la información de depuración es susceptible a la manipulación, la ingeniería inversa y la piratería[89].

Por sí mismo, Java contiene buenas medidas para garantizar la seguridad, por ejemplo, la resistencia implícita a desbordamientos de búfer (buffer overflow) y errores en la de gestión de memoria:

En Java, Todos los accesos a matrices son verificados contra la longitud asignada a la misma matriz. Los Desbordamientos de búfer son atrapados de forma confiable, y desencadenan una excepción, lo cual es mejor porque evita una vulnerabilidad de ejecución remota de código.

La asignación de memoria se gestiona a través de un recolector de basura (Garbage collector), lo cual evita errores del tipo uso después de liberación (user-after-free) y doble liberación (double-free). Además, el Garbage Collector permite

[89] ARXAN. TECHNOLOGIES. Security for Android Java Mobile Applications. [en línea] [citado el 12 Enero, 2015]. Disponible en internet: <https://www.arxan.com/products/mobile/guardit-for-java/>

un manejo más fácil de cadenas de caracteres, lo que elimina en la mayoría de casos, los errores de desbordamiento de búfer.

Java implementa una tipificación estricta, es decir, un código no puede acceder a los bytes de datos por lo que no son. Esto evita vulnerabilidades donde se transgreden los tipos de datos llegando a obtener ClassCastException en tiempo de ejecución.

Estas y otras características hacen de Java un lenguaje de programación mucho más fuerte que C o C++ cuando se trata de seguridad. Pero no lo hacen inmune a las vulnerabilidades de seguridad y a los ataques informáticos.

Algunas de las vulnerabilidades de seguridad de Java son[90]:

- Ingeniería inversa: los atacantes pueden realizar ingeniería inversa a un archivo de código de bytes y descompilar el código después de que se descarga en el cliente. Esto permite robo de propiedad intelectual, también la ingeniería inversa de rutinas de seguridad u otras rutinas importantes que pueden ser explotadas.
- Derivación (Bypass) de rutinas críticas: Los atacantes pueden "parchear" binarios del cliente para evitar la lógica de autenticación o explotar funcionalidades restringidas dentro del código cliente.
- Robo de credenciales y Llaves: muchas veces las claves secretas o credenciales de autenticación están codificados dentro de los componentes, los cuales son muy fáciles de identificar y atacar para obtener dicha información.
- Fácil descompilación: Como un lenguaje interpretado, Java es muy fácil de descomponer. Actualmente existe un amplio número de programas para descompilar el código de bytes y generar código fuente fácil de leer y entender. Los atacantes pueden modificar o falsificar el código, generando versiones "hackeadas" del original.
- Inyección de código SQL: ejecución de una sentencia SQL generado dinámicamente.
- Ataques Cross Site scripting (XSS) en parámetros HTTP a un Servlet. Se presenta cuando en las clases dentro de un servlet no se validan los parámetros HTTP recibidos desde el cliente. El atacante puede enviar código malicioso ejecutable, desde el navegador del cliente, en los parámetros URI o HTTP enviados al servlet y si estos no son validados correctamente se convierten en una vulnerabilidad de seguridad.
- Consulta SQL preparada con parámetros dinámicos: la consulta se realiza con una declaración preparada pero utiliza un parámetro de cadena

[90] KUMAR, Ajitesh. Java - Four Security Vulnerabilities Related Coding Practices to Avoid [en línea] [citado el 12 Enero, 2015]. Disponible en internet: <http://java.dzone.com/articles/java-four-security>

almacenado en un búfer, el cual si no es controlado puede ser modificado por los datos del usuario creando una cadena de inyección de código SQL.

- Almacenamiento directo de Arrays: Cuando se almacenan directamente los arrays, pueden contribuir a que el atacante cambie el objeto array por fuera del programa, haciendo que este se comporte en forma inconsistente.

4.1.20 Problemas de seguridad en C++. C++ es un lenguaje de programación multiparadigma (orientado a objetos y estructurado). [91] Los programa escritos en c++ tienen la ventaja de ser compactos y rápidos, el código es portable, es decir, un programa en c++ podrá ejecutarse en cualquier máquina y sobre cualquier sistema operativo.

Algunas de las vulnerabilidades de seguridad en C++ son[92]:

- Buffer overflow: ocurre cuando se presenta un desbordamiento de memoria por la asignación de un valor que supera los límites definidos para el buffer. Si un atacante logra hacer esto fuera del programa podría manipular las posiciones de memoria de forma arbitraria causando problemas de seguridad.
- Condiciones de tiempo de uso y tiempo de chequeo (TOCTOU): ocurre cuando un programa comprueba el estado de un archivo antes de usarlo, un atacante puede cambiar el estado del recurso (modificarlo, reemplazarlo, borrarlo, etc.) haciendo que el programa realice acciones inválidas cuando el recurso está en un estado inesperado.
- Heap overflow: es un buffer overflow con la diferencia que el ataque se produce sobre la pila de memoria, lo cual representa un grave riesgo para el sistema operativo.
- Inyección de comandos: es una vulnerabilidad que permite la ejecución de código no autorizado desde los comandos del sistema operativo. Se presenta cuando los programas no validan correctamente los parámetros que se usan al invocar funciones del sistema (shell) como system() o exec().
- Format string vulnerabilities: ocurre cuando una función de entrada / salida recibe datos por referencia y construye el mensaje de error en la variable de memoria asignada, el cual no es comprobado para desbordamiento de buffer porque la cadena referenciada se conoce que tiene una longitud de 256 bytes o menos, lo cual puede ser manipulado por un atacante, quien ha ingresado los valores de las variables recibidas por la función, las cuales

[91] STROUSTRUP, Bjarne. An Overview of the C++ Programming Language. The Handbook of Object Technology (Editor: Saba Zamir). CRC Press LLC, Boca Raton. 1999. ISBN 0-8493-3135-8. Disponible en internet: <http://www.stroustrup.com/crc.pdf>

[92] SEACORD, Robert. CERT® C Coding Standard, Second Edition, The: 98 Rules for Developing Safe, Reliable, and Secure Systems. [en línea] [citado el 12 Enero, 2015]. Editorial Adisson Wesley Disponible en internet: <https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=146440541>

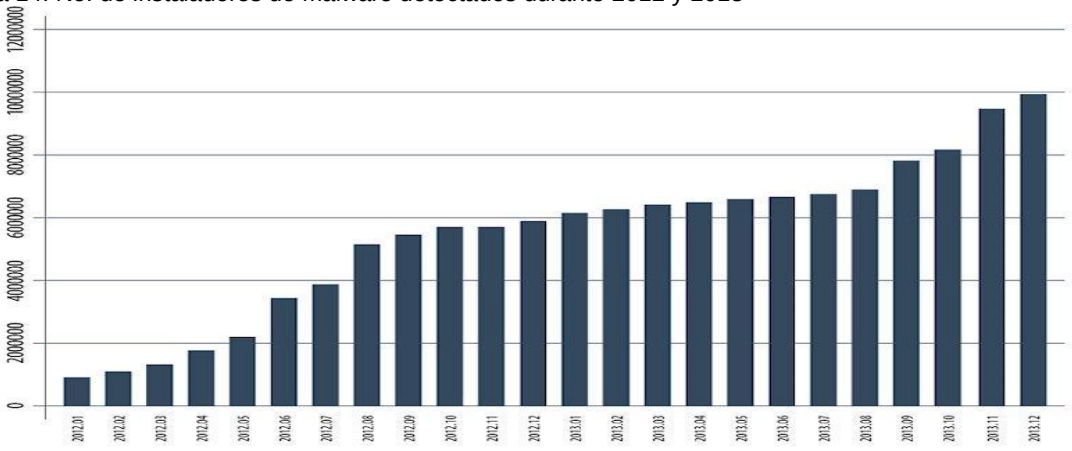
hacen parte del mensaje de salida que permiten controlar total o parcialmente el contenido de la cadena, logrando ver el contenido de la pila, de la memoria o escribir en una ubicación de memoria específica.

- Integer overflow: ocurre cuando se intenta almacenar un valor mayor al valor máximo permitido por el tipo de datos como resultado de una asignación de valor u operaciones entre números enteros, dando lugar a errores fatales o vulnerabilidades que permitan a un atacante sobrescribir directamente en la memoria o el control directo del flujo de ejecución de la aplicación.
- Insecure temporary file: con frecuencia los desarrolladores crean archivos temporales en directorios compartidos utilizados para almacenamiento auxiliar como medio de comunicación con otros procesos para transferencia de datos a través del sistema de archivos, siendo una práctica peligrosa porque el archivo al estar ubicado en un directorio compartido puede ser fácilmente manipulado por un atacante que podría sobrescribir o borrar archivos de la aplicación, logrando una denegación de servicio.

4.2 MARCO CONCEPTUAL

4.2.1 Evolución de la seguridad en los Smartphones. Según estudio realizado por la empresa Kaspersky Lab[93], en el 2013 el malware móvil ha venido aumentando en 3.905.502 millones de instaladores para distribuir *malware* en móviles. Durante los años 2012 y 2013 el software de seguridad de Kaspersky detectó aproximadamente 10 millones como se muestra en la figura 14.

Figura 14. No. de instaladores de malware detectados durante 2012 y 2013

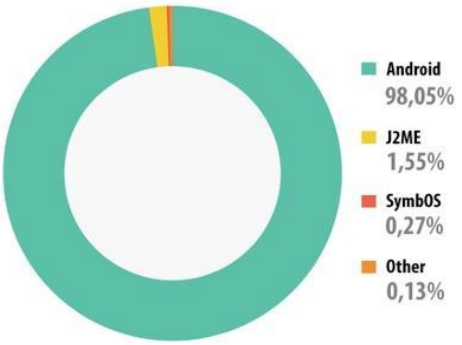


Fuente Kaspersky Lab. Mobile Malware Evolution: 2013. [en línea]. Disponible en internet http://blog.kaspersky.com/mobile-malware-evolution-2013/

[93] KASPERSKY LAB. Mobile Malware Evolution: 2013. [en línea] [citado el 16 mayo, 2014]. Disponible en internet: <http://blog.kaspersky.com/mobile-malware-evolution-2013/>

Android sigue siendo el objetivo primordial del software malicioso en 2013: el 98.05% de los ataques fueron dirigidos a esta plataforma como se ilustra en la figura 15.

Figura 15. Distribución del malware detectado en 2013 por Sistema Operativo



Fuente: Kaspersky Lab. Mobile Malware Evolution: 2013. [en línea]. Disponible en internet: http://blog.kaspersky.com/mobile-malware-evolution-2013/

SophosLabs en su informe[94] describe el incremento de malware durante los últimos 10 años como se ilustra en la figura 16, convirtiéndose en una verdadera amenaza para los usuarios finales debido al rápido crecimiento de los Smartphones, llevando a un aumento inevitable de focalización de estos dispositivos por los ciberdelincuentes.

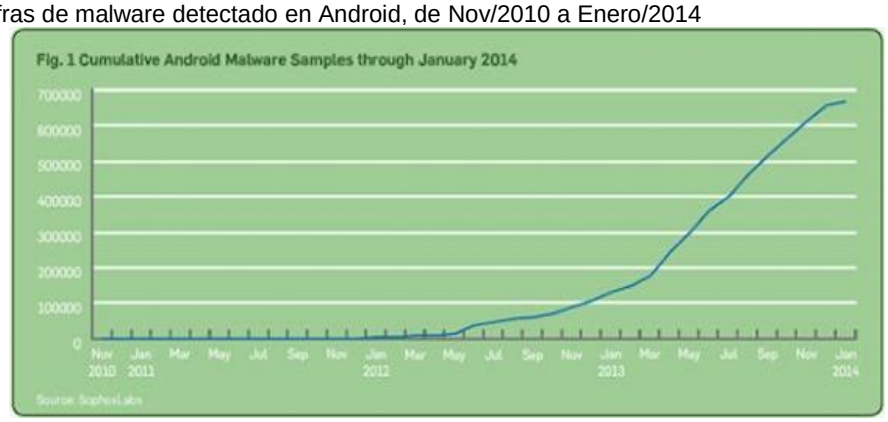
Figura 16. Evolución de amenazas durante los últimos 10 años

Fuente: Sophos. Sophos Mobile Security Threat Report [en línea]. Disponible en internet: http://www.sophos.com/en-us/medialibrary/PDFs/other/sophos-mobile-security-threat-report.pdf

[94] SOPHOS. Sophos Mobile Security Threat Report. Launched at Mobile World Congress, 2014. [en línea] [citado el 16 mayo, 2014]. Disponible en internet: <http://www.sophos.com/en-us/medialibrary/PDFs/other/sophos-mobile-security-threat-report.pdf

El *malware* para Android se ha incrementado llegando a alcanzar más de 650.000 casos. Aunque es una cifra pequeña en comparación con la cantidad de tipos de *malware* que hay para PC con Windows, el software malicioso para Android es la amenaza de más rápido crecimiento para los usuarios. En la figura 17 se ilustra las cifras de malware para Android de Noviembre de 2010 a Enero de 2014.

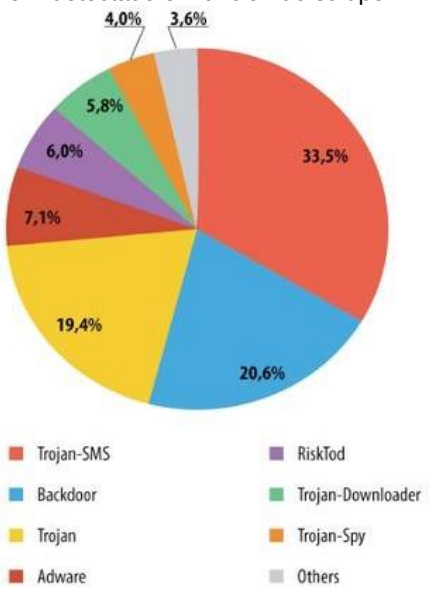
Figura 17. Cifras de malware detectado en Android, de Nov/2010 a Enero/2014



Fuente: Sophos. Sophos Mobile Security Threat Report [en línea]. Disponible en internet: http://www.sophos.com/en-us/medialibrary/PDFs/other/sophos-mobile-security-threat-report.pdf

La mayoría de software malicioso estaba enfocado en el robo menor de dinero mediante llamadas y mensajes. Sin embargo, a lo largo del año, el *malware* móvil diseñado para el *phishing* y robo de información de tarjetas de crédito y dinero aumentó en un 19.7%. En la figura 18 se ilustra la distribución de malware por categoría.

Figura 18. Distribución del malware móvil detectado en función de su tipo



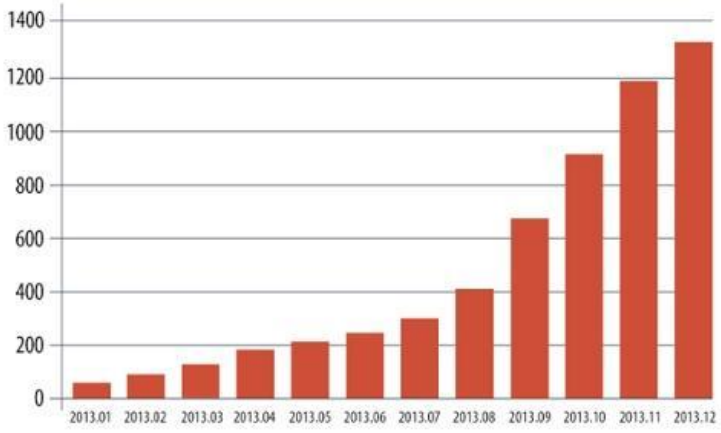
Fuente: Kaspersky Lab. Mobile Malware Evolution: 2013. [en línea]. Disponible en internet: http://blog.kaspersky.com/mobile-malware-evolution-2013/

El principal objetivo de los ciberdelincuentes es conseguir que el software malicioso permanezca en el dispositivo infectado el mayor tiempo posible, para obtener mayores beneficios. Para ello, los criminales escriben virus complejos que son difíciles de encontrar o eliminar. Los esfuerzos en la creación de *malware* se centran en los siguientes factores:

- Las vulnerabilidades de Android: se utilizan para omitir la comprobación de la integridad del código cuando se instala una aplicación (vulnerability Master Key), con el objetivo de fortalecer las aplicaciones maliciosas, haciendo que sean más difíciles de eliminar.
- El código malicioso se integra en programas legítimos para ocultar cualquier signo de infección.
- Ataques en Windows XP: permite al malware móvil infectar el computador después de conectado el Smartphone.

Lo más destacado en 2013 ha sido el rápido aumento de los troyanos de la banca móvil en Android. A principios de año, Kaspersky Lab tenía conocimiento de 67 troyanos bancarios, y a final de año, la cifra ya rondaba los 1321 casos. Este tipo de troyanos tiene como objetivo conseguir beneficios económicos a través de los fraudes bancarios. En la figura 19 se ilustra el aumento de troyanos bancarios móviles.

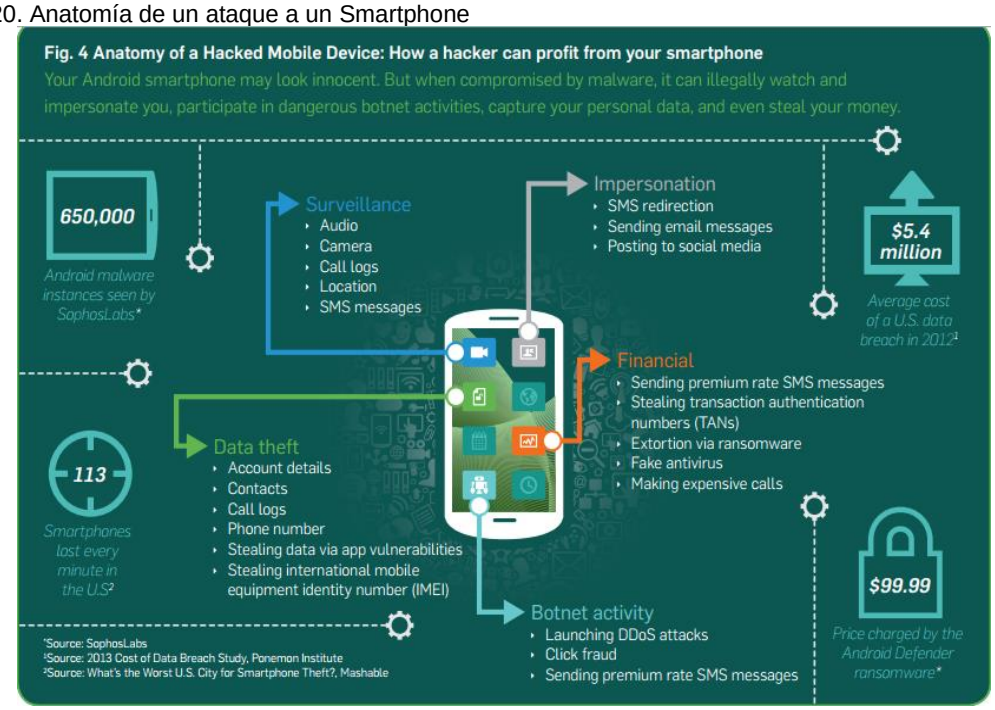
Figura 19. Número troyanos bancarios para móvil contabilizados por Kaspersky



Fuente: Kaspersky Lab. Mobile Malware Evolution: 2013. [en línea]. Disponible en internet: http://blog.kaspersky.com/mobile-malware-evolution-2013/

SophosLabs describe la Anatomía de un Smartphone hackeado como se ilustra en la figura 20, mostrando las diferentes maneras de que un ciberdelincuente puede beneficiarse desde un dispositivo móvil comprometido. Algunos de éstos, tales como ransomware, fake AV, botnet y el robo de datos.

Figura 20. Anatomía de un ataque a un Smartphone



Fuente: Sophos. Sophos Mobile Security Threat Report. Disponible en internet: http://www.sophos.com/en-us/medialibrary/PDFs/other/sophos-mobile-security-threat-report.pdf

4.3 MARCO CONTEXTUAL

A study of Android Application Security[95]

William Enck, Damien Octeau, Patrick McDaniel, and Swarat Chaudhuri publicaron "*A Study of Android Application Security*" en el 20° USENIX Security Simposium del 2011. El trabajo buscó comprender la seguridad de aplicaciones del Smartphone mediante el estudio de las 1.100 aplicaciones gratis más populares de Android. Para lograr el análisis se utilizó el decompilador DED, que recupera el código fuente de la aplicación Android directamente de su imagen de instalación; los investigadores realizaron un estudio basados en el análisis de 21 millones de líneas de código recuperadas. El análisis descubrió uso generalizado o el mal uso de datos de identificación personal, datos de identificación del teléfono y una presencia masiva de redes de publicidad.

Review on Android and Smartphone Security[96]

Tiwari Mohini, Srivastava Ashish Kumar and Gupta Nitesh publicaron "*Review on Android and Smartphone Securtity*" en el Research Journal of Computer and Information Technology Sciences en el 2013. El trabajo buscó comprender la plataforma y el modelo de seguridad del sistema operativo Android revisando los casos de seguridad de Smartphone basados en Android y que el uso de la tecnología en la certificación de aplicaciones implica unos retos técnicos y logísticos más complejos.

Dissecting Android Malware: Characterization and Evolution[97]

Yajin Zhou y Xuxian Jiang publicaron "*Dissecting Android Malware: Characterization and Evolution*" en mayo de 2012. Realizaron un estudio para caracterizar el malware existente para Android, recogiendo más de 1200 muestras de malware desde agosto de 2010 a octubre de 2011 con el fin de analizarlas, compararlas y clasificarlas para tener un mejor conocimiento del malware actual y observar cómo ha ido evolucionando desde su aparición.

[95] ENCK, William, et al. A study of Android Application Security . [en línea] [citado el 12 noviembre, 2014]. Disponible en internet: <http://www.cs.rice.edu/~sc40/pubs/enck-sec11.pdf>

[96] MOHINI, Tiwari, et al. Review on Android and Smartphone Security. [en línea] [citado el 8 diciembre, 2014]. Disponible en internet: <http://www.academia.edu/6375839/Review_on_Android_and_Smartphone_Security>

[97] ZHOU, Yajin y Xuxian Jiang. Android Malware Genome Project. [en línea] [citado el 8 diciembre, 2014]. Disponible en internet: <http://www.malgenomeproject.org/

4.4 MARCO LEGAL

4.4.1 Colombia. El congreso de la Republica de Colombia con la promulgación de la Ley 599 de Julio 24 de 2000, “por la cual se expide el Código Penal”[98] en su Capítulo séptimo del Libro segundo, del Título III: Delitos contra la libertad individual y otras garantías, trata sobre la violación a la intimidad, reserva e interceptación de comunicaciones en el artículo 192: Violación ilícita de comunicaciones, permitiendo que se diera inicio a la protección de los datos en Colombia.

Con el auge que ha tenido el internet y el rápido crecimiento de los medios informáticos, ha llevado a la masificación del uso de dispositivos móviles y por consiguiente el aumento de los delitos informáticos, llevando a que el Congreso de la Republica de Colombia sancionará la Ley 1273 el 5 de enero de 2009, "*por medio del cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado –denominado 'De la Protección de la información y de los datos'– y se preservan integralmente los sistemas que utilicen las tecnologías de la información y las comunicaciones, entre otras disposiciones*"[99] con el fin de establecer normatividad para combatir el cibercrimen.

Con la creación de esta Ley se da un valor jurídico a la información, estableciendo las conductas criminales que tienen que ver con sistemas de cómputo y las nuevas tecnologías.

En esta ley se encuentran establecidos artículos para los diferentes ciberdelitos, los cuales son aplicables a la presente investigación:

- Artículo 269A: *Acceso abusivo a un sistema informático*: ocurre cuando el ciberdelincuente aprovecha la vulnerabilidad en el acceso al sistema para extraer beneficios económicos o personales.

- Artículo 269B: *Obstaculización ilegítima de sistema informático o red de telecomunicación:* ocurre cuando el ciberdelincuente bloquea de forma ilegal el uso de un sistema hasta cuando obtiene un beneficio económico.

- Artículo 269C: *Interceptación de datos informáticos*: cuando valiéndose de los recursos tecnológicos, sin autorización legal obstruye datos.

[98] SENADO DE LA REPUBLICA. Ley 599 de 2000. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: <http://www.secretariasenado.gov.co/senado/basedoc/ley_0599_2000.html>
[99] SENADO DE LA REPUBLICA. Ley 1273 de 2009. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: <http://www.secretariasenado.gov.co/senado/basedoc/ley_1273_2009.html>

- Artículo 269D: *Daño Informático*. cuando una persona que sin estar autorizada, modifica, altera, daña, borra, destruye o suprime datos del programa o de documentos electrónicos.

- Artículo 269E: *Uso de software malicioso*: cuando se producen, adquieren, venden, distribuyen, envían, introducen o extraen del país software que produce daños en los recursos informáticos.

- Artículo 269F: *Violación de datos personales:* cuando un individuo sin estar facultado, sustrae, vende, envía, compra, divulga o emplea datos personales almacenados con el fin de lograr utilidad personal o para otros.

4.4.2 Tratados de Cooperación internacional. Colombia a finales de septiembre de 2013 fue invitada por el Consejo de Europa a adherirse a la convención sobre delito cibernético o Convención de Budapest para hacer parte de los tratados de asistencia mutua, con el fin de participar de manera cooperativa en asuntos de delitos transnacionales y en investigaciones criminales globales[100].
El Convenio de Budapest se presenta como una solución internacional existente para atacar los delitos informáticos y/o electrónicos, convirtiéndose en una herramienta para la armonización legislativa internacional y su lucha contra el ciberdelito". Así mismo, "el Convenio sobre la Ciberdelincuencia del Consejo de Europa es el único acuerdo internacional que cubre todas las áreas relevantes de la legislación sobre ciberdelincuencia (derecho penal, derecho procesal y cooperación internacional). Adoptado por el Comité de Ministros del Consejo de Europa en su sesión N. 109 del 8 de noviembre de 2001, se presentó a firma en Budapest, el 23 de noviembre de 2001 y entró en vigor el 1 de julio de 2004.[101]

El Convenio de Budapest contiene la siguiente clasificación de los tipos de delitos, organizados en cuatro vectores:

a. Delitos contra la confidencialidad, la integridad y la disponibilidad de los datos y los sistemas informáticos.
b. Delitos informáticos.
c. Delitos relacionados con el contenido, donde se encuentran delitos relacionados con la pornografía infantil.
d. Delitos relacionados con infracciones de la propiedad intelectual.

[100] MARRUGO, Iván. Colombia y la cooperación internacional en los delitos informáticos. [en línea] [citado el 15 diciembre, 2014]. Disponible en internet: <http://www.ambitojuridico.com/BancoConocimiento/N/noti-140130-06colombia_y_la_cooperacion_internacional_en_los_delitos_inform/noti-140130-06colombia_y_la_cooperacion_internacional_en_los_delitos_inform.asp?IDObjetoSE=17572>
[101] CONCIL OF EUROPE. Convenio sobre la Ciberdelincuencia. Budapest, 23.XI.2001. [en línea] [citado el 15 diciembre, 2014]. Disponible en internet: http://www.coe.int/t/dghl/cooperation/economiccrime/Source/Cybercrime/TCY/ETS_185_spanish.PDF

4.4.3 Google. En las condiciones del servicio de Google Inc[102], con última modificación el 30 de Abril de 2014, establece que al utilizar los servicios se aceptan las políticas y condiciones establecidas e informan que sus servicios usan cierto contenido que no pertenece a la empresa, el cual es responsabilidad del proveedor que lo pone a disposición. Google puede revisar si el contenido publicado es ilegal o infringe las políticas o la ley, eliminándolo o rechazándolo.

A su vez Google establece un Acuerdo de distribución para desarrolladores[103] de Google Play en lo cual se puede resaltar los siguientes aspectos:

- El Desarrollador acepta utilizar Google Play Store solo para los fines permitidos por el Acuerdo y cualquier ley, normativa, norma o práctica de aceptación general aplicable existente en las jurisdicciones correspondientes.

- El uso de Google Play Store por parte del desarrollador es para distribuir Productos, proteger los derechos legales y la privacidad de los usuarios en cuanto a nombres de usuario, contraseñas u otra información personal o de inicio de sesión, por lo cual debe informar a los usuarios que su información estará disponible para su Producto, proporcionándole un aviso de privacidad legalmente válido, así como la protección correspondiente.

- El Producto del Desarrollador solo podrá utilizar la información de usuarios para los fines específicos que previamente se le ha concedido permiso. Cuando el producto almacena información personal o confidencial proporcionada por los usuarios, debe hacerlo de un modo suficientemente seguro y únicamente durante el tiempo necesario. No obstante, si el usuario ha suscrito un acuerdo independiente con el Desarrollador que permite que el Desarrollador o su Producto almacenen o utilicen información personal o confidencial directamente relacionada con su Producto, el uso de dicha información se regirá por las condiciones de ese acuerdo independiente.

4.4.4 Estados Unidos. En los Estados Unidos, existen leyes federales que protegen contra el ataque a ordenadores, uso ilegítimo de passwords, invasiones electrónicas en la privacidad, y otras transgresiones.

La ley Federal de EEUU más importante utilizada para perseguir a los delincuentes informáticos es el Acta de Abuso y Fraude Computacional CFAA de 1994 que modificó al Acta de Fraude y el Acta Federal de Abuso Computacional

[102] GOOGLE INC. Condiciones del Servicio de Google. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: <https://www.google.com.co/intl/es-419/policies/terms/regional.html>
[103] GOOGLE INC. Condiciones del Servicio de Google. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: <https://play.google.com/intl/ALL_es/about/developer-distribution-agreement.html>

de 1986. Que para la presente investigación se enunciará el Titulo 18 U.S.C. que detalla “Delitos y procedimiento penal”, capítulo 47, Secciones 1029[104] y 1030[105].

Sección 1029[106]

La sección 1029 relativa a "*Fraude y actividades relacionadas en conexión con dispositivos de acceso*", establece que quien produce, vende o utiliza dispositivos de acceso falsificados o instrumentos de telecomunicaciones con la intención de cometer fraude y obtener servicios o productos con un valor de US$ 1000, infringe la ley. Tipifica como delito el uso indebido de contraseñas de computadoras y otros dispositivos de acceso como las tarjetas token.

Prohíbe el fraude y cualquier actividad relacionada que pueda realizarse mediante el acceso o uso de dispositivos falsificados como PINs, tarjetas de crédito, números de cuentas y algunos tipos más de identificadores electrónicos.

Las nueve áreas de actividad criminal que se cubren en esta sección se describen a continuación. Todas “requieren” que el delito implique comercio interestatal o con el extranjero.

1. Producción, uso o tráfico de dispositivos de acceso falsificados.
2. Uso u obtención sin autorización de dispositivos de acceso para obtener algo de valor totalizando $1000 o más, durante un periodo de un año.
3. Posesión de 15 o más dispositivos de acceso no autorizados o falsificados.
4. Fabricación, tráfico o posesión de equipo de fabricación de dispositivos de acceso ilegales.
5. Realización de transacciones con dispositivos de acceso pertenecientes a otra persona con el objetivo de obtener dinero o algo de valor totalizando $1000 o más durante un periodo de un año.
6. Solicitar a una persona con el objetivo de ofrecerle algún dispositivo de acceso o venderle información que pueda ser usada para conseguir acceso a algún sistema.
7. Uso, producción, tráfico o posesión de instrumentos de telecomunicación que hayan sido alterados o modificados para obtener un uso no autorizado de un servicio de telecomunicaciones.
8. Uso, fabricación, tráfico o posesión de receptores-escaneadores o hardware o software usado para alterar o modificar instrumentos de telecomunicaciones para obtener acceso no autorizado a servicios de

[104] U.S. GOVERNMENT PRINTING OFFICE. §1029. Fraud and related activity in connection with access devices. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: <http://www.gpo.gov/fdsys/pkg/USCODE-2010-title18/html/USCODE-2010-title18-partI-chap47-sec1029.htm>

[105] U.S. GOVERNMENT PRINTING OFFICE. §1030. Fraud and related activity in connection with computers. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: <http://www.gpo.gov/fdsys/pkg/USCODE-2010-title18/html/USCODE-2010-title18-partI-chap47-sec1030.htm>

[106] U.S. GOVERNMENT PRINTING OFFICE Op. Cit.

telecomunicaciones. Esto también incluye los escáneres que mucha gente usa para interceptar llamadas de teléfonos celulares (hammers).

9. Hacer creer a una persona que el delincuente es un miembro de su compañía de tarjeta de crédito o su agente para obtener dinero o realización de transacciones hechas con un dispositivo de acceso y viceversa (tratar de hacer creer a la compañía de crédito que se trata de la persona legítima).

Sección 1030[107]

La sección 1030 relativa a "*Fraudes y actividades relacionadas con conexión de computadoras*", prohíbe el acceso a computadoras protegidas sin autorización para causar daño. Esta ley tipifica como delito la propagación de virus, gusanos y el acceso a los sistemas informáticos por personas no autorizadas.

Esta sección como parte de la Ley sobre Abuso y Fraude Informático de 1986, prohíbe el acceso no autorizado o fraudulento a ordenadores gubernamentales, y establece diversas condenas para esa clase de accesos. Bajo la Ley de Abuso y Fraude Informático, el Servicio Secreto americano y el F.B.I. tienen jurisprudencia para investigar los delitos definidos en este decreto.
Las seis áreas de actividad criminal cubiertas son:

1. Adquisición de información restringida relacionada con defensa nacional, asuntos exteriores o sobre energía nuclear con el objetivo o posibilidad de que sean usados para dañar a los Estados Unidos o para aventajar a cualquier otra nación extranjera.

2. Obtención de información en un registro financiero de una institución fiscal o de un propietario de tarjeta de crédito, o de información de un cliente en un archivo de una agencia de información de clientes.

3. Atacar un ordenador que sólo corresponda ser usado por algún departamento o agencia del gobierno de los EEUU, para el caso de que no sólo puede ser usada por esta agencia, atacar un ordenador usado por el gobierno en el que la intrusión producida afecte el uso que el gobierno hace de él.

4. Promover un fraude accediendo a un ordenador de interés federal y obtener algo de valor, a menos que el fraude y la cosa obtenida consistan solamente en el uso de dicho ordenador.

[107] U.S. GOVERNMENT PRINTING OFFICE. Op. cit.

5. A través del uso de un ordenador utilizado en comercio interestatal, transmitir intencionadamente programas, información, códigos o comandos a otro sistema informático. Existen dos situaciones diferentes:

    A- En esta situación (I) la persona que realiza la transmisión está intentando dañar el otro ordenador o provocar que no se permita a otras personas acceder a él; y (II) la transmisión se produce sin la autorización de los propietarios u operadores de los ordenadores, y causa $1000 o más de pérdidas, o modifica o perjudica, o potencialmente modifica o altera un examen o tratamiento médico.

    B- En esta situación, (I) la persona que realiza la transmisión no intenta hacer ningún daño, pero actúa imprudentemente despreciando el riesgo que existe de que la transmisión causara daño a los propietarios u operadores de los ordenadores y provoca $1000 o más de pérdidas, modifica o potencialmente modifica un examen o tratamiento médico.

6. Promover el fraude traficando con passwords o información similar que haga que se pueda acceder a un ordenador sin la debida autorización. Todo esto si ese tráfico afecta al comercio estatal o internacional o si el ordenador afectado es utilizado por o para el Gobierno. (El delito debe ser cometido conscientemente y con voluntad de estafar.)

5. DISEÑO METODOLÓGICO PRELIMINAR

Para el desarrollo de la presente investigación, guiados por el planteamiento del problema y en busca de alcanzar los objetivos planteados, se definen los siguientes componentes:

5.1 TIPO DE INVESTIGACIÓN

La presente investigación es Descriptiva, porque el objetivo del estudio es describir las vulnerabilidades de las aplicaciones móviles nativas para el sistema operativo Android, versión Jelly Bean 4.1.2 enfocado a los riesgos OWASP Mobile M2 y M3 y de tipo experimental porque se realizaran pruebas a las aplicaciones móviles nativas.

5.2 POBLACIÓN

La población es delimitada a las aplicaciones móviles nativas para el sistema operativo Android versión 4.1.2 Jelly Bean en dispositivos móviles Smartphone.

5.3 MUESTRA

Para cumplir con los objetivos trazados en esta investigación, la muestra extraída es de tipo no probabilística, dirigida y de carácter intencionado seleccionando las diez (10) mejores aplicaciones móviles nativas del 2014 consideradas por Google Play Store, las cuales fueron instaladas en un dispositivo móvil Smartphone con sistema operativo Android versión 4.1.2 Jelly Bean y/o en el emulador de Android con la versión seleccionada.

5.4 TÉCNICAS PARA LA RECOLECCIÓN DE DATOS

A continuación se describe la forma en que se obtuvo la información para el desarrollo de la investigación.

5.4.1 Técnicas de recolección de información. Para la recolección de la información se ha usado:

- Documentos (medios impresos, grabaciones de audio y video, documentación en medios electrónicos) sobre la arquitectura del sistema operativo Android y se esquema de seguridad y proyecto OWASP mobile.
- Investigación de las herramientas utilizadas para analizar aplicaciones móviles Android. Se revisaron técnicas de ingeniería inversa, descompiladores, desensambladores, funcionamiento del SDK.
- Resultados obtenidos de las pruebas de hacking ético (análisis dinámico, análisis estático) realizadas a las aplicaciones instaladas en un dispositivo móvil y/o emulador de tipo Smartphone con sistema operativo Android.

5.4.2 Fuentes de información

5.4.2.1 Fuentes primarias. Se utilizarán datos obtenidos de búsqueda bibliográfica, artículos científicos, monografías, tesis, libros o artículos de revistas especializadas originales, no interpretados referentes a la seguridad en las aplicaciones móviles nativas en el sistema operativo Android, objeto del estudio.

5.4.2.2 Fuentes secundarias. Se utilizará resúmenes, compilaciones o listados de referencias, preparados en base a las fuentes primarias sobre herramientas para análisis de aplicaciones móviles.

5.4.3 Técnicas de procesamiento y análisis de datos. Para el estudio de las fuentes obtenidas, se utilizará un análisis de la información documental, pruebas de penetración en dispositivo móvil o emulador, para así poder presentar, un análisis crítico a acerca del resto de la información obtenida en la recolección de datos, puesto que la misma fue relevante para el desarrollo del tema de investigación.

5.5 ACTIVIDADES

Se realizaron las siguientes etapas:

Etapa 1. Levantamiento de Información. Recolección de la información para aproximarnos al tema usando fuentes primarias o registro secundario de tipo bibliográfico tanto en textos elaborados, documentos electrónicos escritos por expertos en la materia. Se realizaron las siguientes actividades:

- Revisión y documentación de la arquitectura y modelo de seguridad del sistema operativo Android.
- Identificación de vulnerabilidades de seguridad en dispositivos Smartphone con sistema operativo Android versión Jelly Bean 4.1.2 a nivel de aplicaciones móviles nativas en enfocado a los riesgos OWASP Mobile M2 y M3.
- Selección de las aplicaciones móviles nativas para la ejecución de las pruebas.

Etapa 2: Interpretación y análisis de la información. Se realizaron las siguientes actividades:

- Configuración del entorno de trabajo para la realización de las pruebas a las aplicaciones seleccionadas.
- Realización de pruebas de hacking ético a las aplicaciones instaladas en el emulador o dispositivo móvil Smartphone con sistema operativo Android versión 4.1.2 Jelly Bean, siguiendo el Plan de pruebas diseñado y usando las herramientas escogidas para el desarrollo de las pruebas de hacking ético.
- Construcción del informe del proceso de evaluación de seguridad de las aplicaciones móviles nativas analizadas de acuerdo a los resultados obtenidos.
- Construcción del documento final del proyecto de investigación.

Etapa 3: Entrega del Documento final. Presentación de análisis final de resultados obtenidos a manera de conclusiones, presentación del informe de evaluación de las aplicaciones móviles nativas analizadas y presentación formal del documento final de la investigación.

6. RESULTADOS

6.1 APLICACIONES SELECCIONADAS

Para el desarrollo de la presente investigación fueron seleccionadas diez de las mejores aplicaciones gratuitas del año 2014 según Google Play Store[108]; estas aplicaciones serán sometidas a diversas pruebas de seguridad con el fin de identificar posibles vulnerabilidades que puedan afectarlas, poniendo en riesgo la seguridad de la información almacenada en los dispositivos en los cuales se encuentran instaladas.

A continuación se caracteriza cada una de las aplicaciones seleccionadas:

Tabla 2. Aplicaciones móviles objeto de estudio

Aplicación	**Wunderlist** (com.wunderkinder.wunderlistandroid)			
	Es una aplicación que permite crear listas de tareas y sincronizarlas entre diferentes dispositivos y compartir listas con otros usuarios. Principales características: trabajo colaborativo, creación listas públicas, creación listas inteligentes, sincronización de la aplicación con Facebook y otras redes. Permite el registro con la cuenta de Facebook, Google o el registro directo en la aplicación mediante el correo electrónico y contraseña.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
3.1.0	4,4 estrellas	6 Wunderkinder GmbH	Free y Pro	20.7 MB
Aplicación	**TED** (com.ted.android)			
	Es la aplicación oficial de TED que permite presentar debates, charlas o discursos en vídeo o audio (TEDTalk) de las personas con más influencia y expertos sobre diferentes temas. Principales características: totalmente traducido y localizado en 21 idiomas, permite añadir subtítulos de acuerdo al idioma, los TEDTalk son actualizados semanalmente.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
2.3.1	4,5 estrellas	TED Conference	Free	7.5 MB
Aplicación	**Teclado SwiftKey + Emoji** (com.touchtype.swiftkey)			

[108] GOOGLE PLAY. Mejores aplicaciones de 2014. [en línea] [citado el 10 Marzo, 2015]. Disponible en internet: <https://play.google.com/store/apps/collection/promotion_3000f13_best_of_2014>

	Es una aplicación que ofrece la mejor predicción de la siguiente palabra, autocorrección inteligente, compatibilidad con más de 800 emoticonos, predicción de emoticonos, escritura rápida de SMS, chat, texto y correo electrónico. Principales características: escritura inteligente y rápida, emoticones y predicción de emoticones, autocorrección inteligente			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
5.2.2.126	4,5 estrellas	SwiftKey	Free	28.0 MB
Aplicación	**Lumosity** (com.lumoslabs.lumosity)			
	Es una herramienta educativa diseñada por científicos para entrenamiento cerebral. Principales características: desarrollo de habilidades cognitivas, ejercita la memoria y atención, prueba de habilidades matemáticas. Permite el registro con la cuenta de Facebook o el registro directo en la aplicación mediante el correo electrónico y contraseña.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
1.0.202	4,0 estrellas	Lumos Labs, Inc	free y premium	45.4 MB.
Aplicación	**Wish** (com.contextlogic.wish)			
	Es una aplicación que permite encontrar y adquirir productos (muebles, joyas, ropa, cosmético, etc.) a precios muy ventajosos. Principales características: los productos se encuentran clasificados por categorías, crear listas de favoritos, obtener información del vendedor y las opiniones de los compradores.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
3.9.0	4,4 estrellas	Wish Inc.	Free	9.4 MB
Aplicación	**Shazam** (com.shazam.android)			
	Es una aplicación para identificar música y programas de televisión, permitiendo escuchar fragmentos de canciones, compartir en redes sociales y realizar compras en Amazon o Google play. Principales características: permite conocer el título, autor y álbum de la canción que está sonando, ofrece la opción de comprar las canciones en iTunes y ver el videoclip.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
5.0.0-14112017	4,4 estrellas	Shazam Entertainment Limited	Free y Premium.	11,0 MB
Aplicación	**IF by IFTTT** (com.ifttt.ifttt)			

	Aplicación que permite crear conexiones potentes con una sencilla premisa “si ocurre esto, haz aquello” Principales características: activa y conecta canales (como, por ejemplo, Facebook, Dropbox y Gmail, así como dispositivos como el Termostato Nest, Fitbit y Hue de Phillips) para disponer de formas ilimitadas de automatizar y empoderarse.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
1.1.2	4,1 estrellas	IFTTT	Free	6.6 MB
Aplicación	**Groupon (**com.groupon**)**			
	Es una aplicación para comprar por internet productos o servicios que se encuentra con descuento, permitiendo obtener un ahorro significativo. Principales características: permite comprar, administrar y canjear los groupones, navegar y acceder a las ofertas del día de la ciudad de ubicación, comprobar y comprar los descuentos de todas las ciudades donde groupon está presente, acceder a la cuenta de groupon para verificar y gestionar compras anteriores.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
7.3797	4,5 estrellas	Groupon, Inc.	Free	14.7 MB
Aplicación	**Locket Lock Screen for English** (com.locket.matterhorn)			
	Es una aplicación de pantalla de bloqueo inteligente que ayuda a aprender inglés a través de historias de tendencias que se muestran cada vez que se desbloquea el dispositivo móvil. Principales características: permite personalizar la pantalla de bloqueo con una configuración hasta de 5 pantallas con las noticias seleccionadas y la posibilidad de guardarlas.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
2.1.17	3,9 estrellas	Locket	Free	6.8 MB
Aplicación	**Timehop** (com.timehop)			
	Timehop es una aplicación que permite recuperar estados, fotos y mensajes de Facebook, Instagram, Twitter y Foursquare y reproduce en el dispositivo móvil pasado un día a la vez. Principales características: requiere login de Facebook y una vez en la aplicación puede vincularse con las cuentas de Twitter, Instagram y Foursquare.			
Versión	**Calificación**	**Desarrollador**	**Licencia**	**Tamaño**
1.4.33	4,0 estrellas	Timehop	Free	5.2 MB

Fuente: Los Autores

6.2 CONFIGURACIÓN DEL AMBIENTE DE PRUEBAS

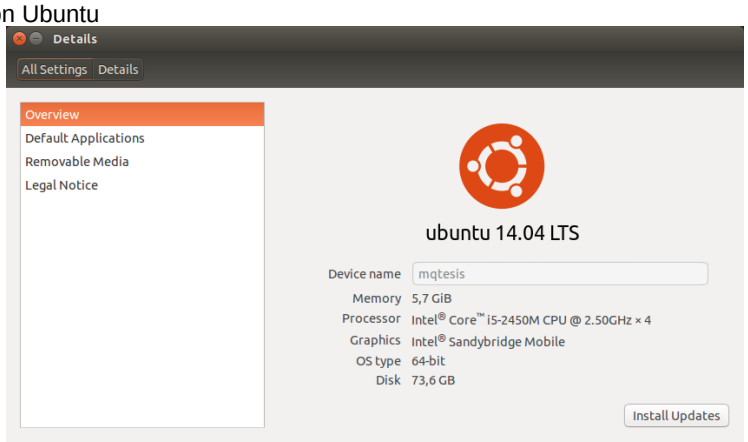
El objetivo de la configuración del ambiente de pruebas es proveer el hardware y software necesario para realizar la ejecución de las pruebas de las aplicaciones en escenarios que cumplan con los requisitos necesarios para su funcionamiento normal con el fin de obtener resultados acertados.

A continuación se describe la configuración, de hardware y software, del ambiente para la ejecución de las pruebas de pentesting realizadas a las aplicaciones móviles nativas utilizadas para el desarrollo del proyecto:

1. Los dos equipos utilizados para realizar las pruebas tienen las siguientes características: procesador Intel Core I5 (2,5 Ghz), memoria RAM de 6GB, Disco duro de 80 GB, tarjeta de red inalámbrica.

2. Para el ambiente de pruebas se seleccionó como sistema operativo base la distribución Linux Ubuntu 14.04 LTS de 64 bits. Descarga realizada de la página oficial http://www.ubuntu.com/download

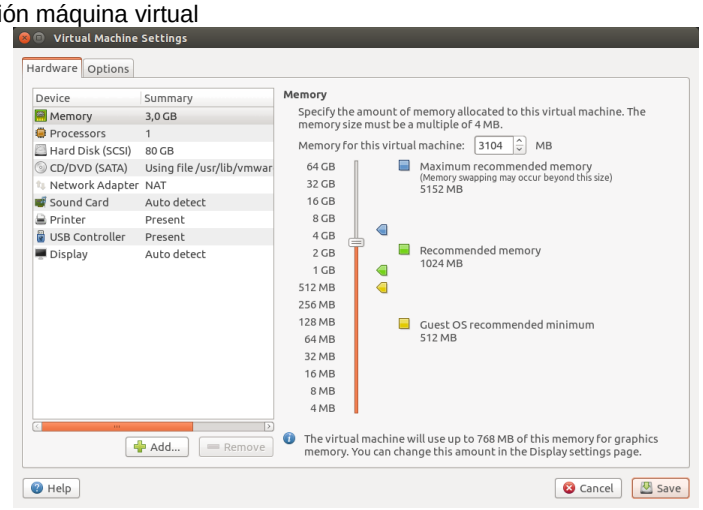
Figura 21. Instalación Ubuntu



Fuente: Los Autores

3. Sobre el sistema operativo Ubuntu se instaló vmware player, versión 7.0.0, para usar una máquina virtual donde se ejecuten las herramientas de pruebas. La máquina virtual tiene la siguiente configuración: procesador de 1 Core, 3GB de memoria RAM, 80GB de disco duro. VMWare player fue descargado de la página oficial del fabricante https://www.vmware.com/co/products/player.

Figura 22. Configuración máquina virtual



Fuente: Los Autores

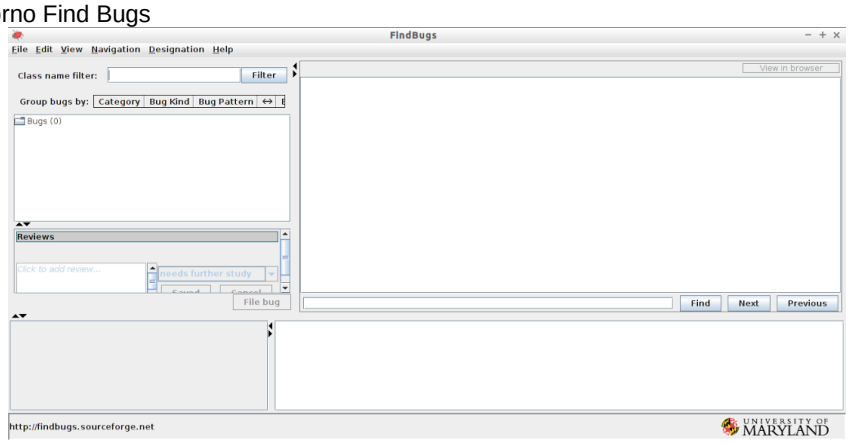
4. En la máquina virtual se instaló la distribución Linux Santoku 0.5, basada en OWASP Mobisec, la cual se especializa en pruebas de seguridad, análisis de malware y análisis forenses para teléfonos móviles, válida para dispositivos con sistema Android; esta distribución incluye las herramientas para realizar Test de penetración a las aplicaciones móviles y el Android SDK Manager para emular el dispositivo Android. Descarga realizada de la página oficial www.nowsecure.com o https://santoku-linux.com

Figura 23. Entorno Distribución Santoku

Fuente: Los Autores

5. Instalación y configuración de Find Bugs, programa que permite realizar análisis de código Java de aplicaciones. Descarga realizada del repositorio oficial http://findbugs.sourceforge.net/

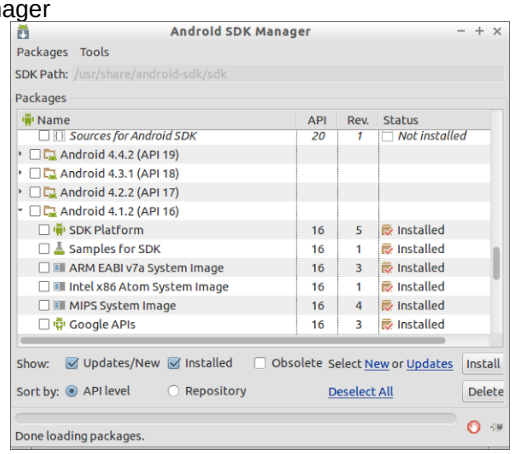
Figura 24. Entorno Find Bugs



Fuente: Los Autores

6. En la máquina virtual fue necesario actualizar Android SDK Manager con la instalación del paquete de Android 4.1.2 (API 16) que corresponde al sistema operativo seleccionado para la investigación.

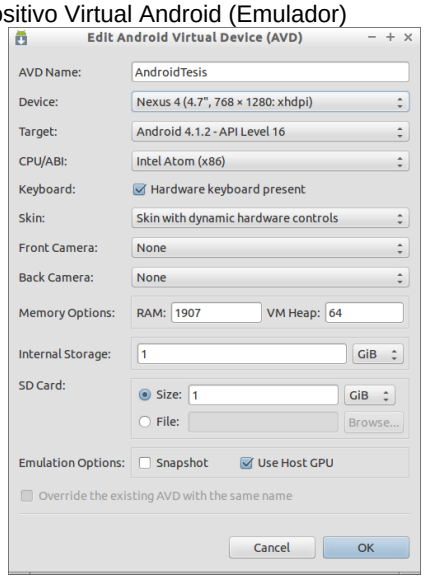
Figura 25. Android SDK Manager



Fuente: Los Autores

7. Se procede a crear y configurar el emulador del dispositivo Android usando la herramienta del Android SDK manager con la siguiente configuración:
   - Nombre AVD: AndroidTesis
   - Dispositivo: Nexus 4 (4.7”, 768 x 1280: xhdpi)
   - Target: Android 4.1.2 – API Level 16
   - CPU/ABI: Intel Atom (x86)
   - Memoria: 2Gb RAM, VM Heap: 64
   - Internal Storage: 1Gb
   - SD Card: 1Gb

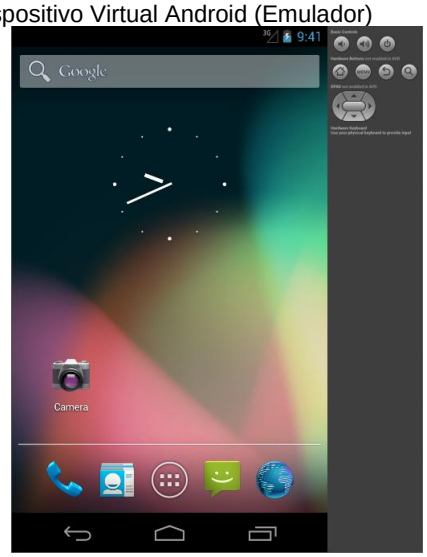
Figura 26. Configuración Dispositivo Virtual Android (Emulador)



Fuente: Los Autores

Se realiza el arranque del emulador, el cual se ejecuta sin errores; adicional se realiza navegación por el mismo con el fin de verificar su funcionamiento.

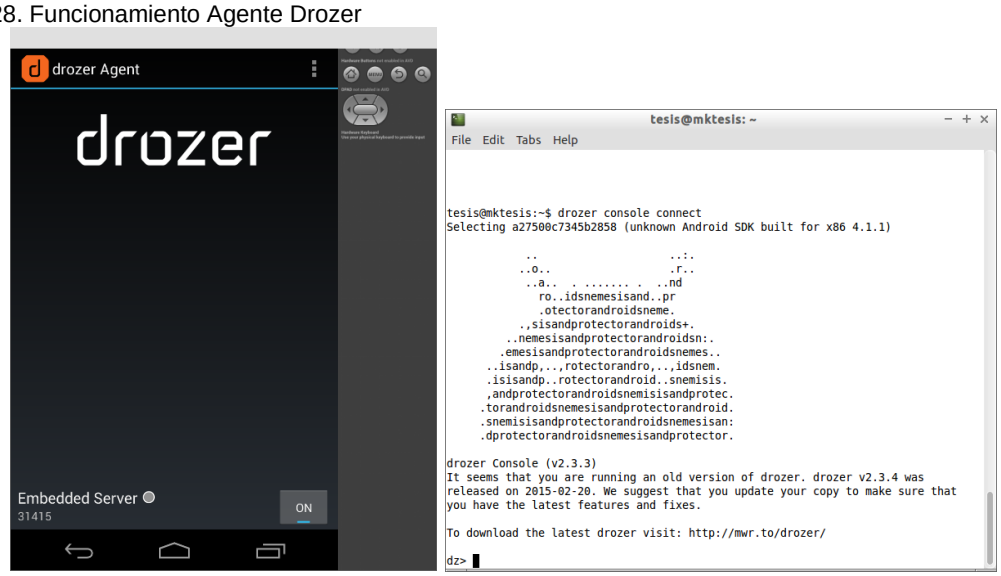
Figura 27. Funcionamiento Dispositivo Virtual Android (Emulador)



Fuente: Los Autores

8. Instalación del agente Drozer versión Community Edition. Descarga realizada de la página oficial de MWR Infosecurity https://www.mwrinfosecurity.com/products/drozer/community-edition/

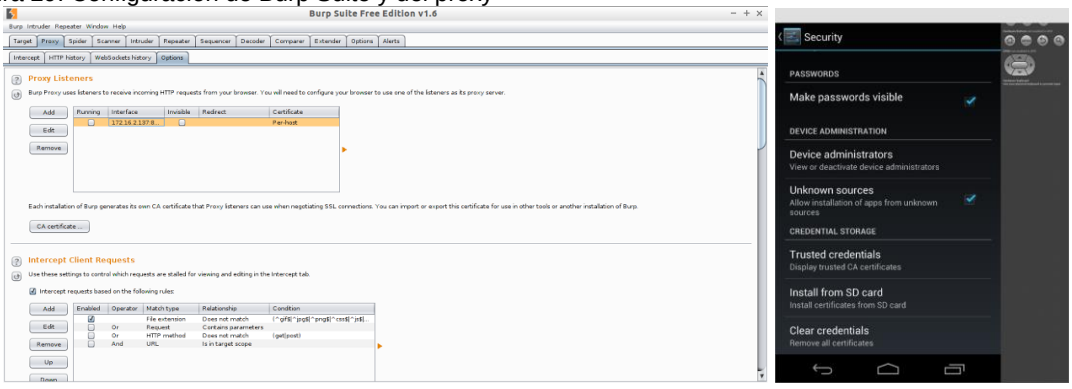
Figura 28. Funcionamiento Agente Drozer



Fuente: Los Autores

9- Configuración de Burp Suite con la dirección IP del equipo anfitrión y configuración del emulador para que navegue a través del proxy e instalación del certificado de PortSwigger.

Figura 29. Configuración de Burp Suite y del proxy



Fuente: Los Autores

6.3 HERRAMIENTAS SELECCIONADAS PARA REALIZAR LAS PRUEBAS

Para la realización de las pruebas de las aplicaciones móviles fueron seleccionadas las siguientes herramientas:

Tabla 3. Herramientas seleccionadas

Herramienta	Descripción	Licencia
APK Downloader	Web service para descargar aplicaciones	Free
ADB (Android Debug Bridge)	Herramienta de línea de comandos para comunicación con el emulador o dispositivo Android	Free
Dex2Jar	Convierte .DEX a código .Jar (Java)	Free
JD-GUI	Lee código .Jar	Free
Androguard	Descompilación (Dex/Odex, apk, xml binario a xml android, recursos arsc)	Free
Apktool	Descompilación (Dex a Smali)	Free
Eclipse	Es un IDE muy completo y adaptable, que permite configurar el ambiente de desarrollo y vincular plugins como módulos independientes que brindan un enfoque robusto para desarrollos JEE.	Free
Burp Suite	Herramienta para realizar test de penetración, que puede ser usada como un proxy para interceptar el tráfico de la red.	Free
Find Bugs	Es una herramienta de código abierto para análisis estático que busca errores en el código java.	Free
Drozer	Es un framework para Android para realizar análisis de vulnerabilidades de aplicaciones y dispositivos móviles.	Free
Sqliteman	Es una aplicación que gestiona bases de datos SQLite desde una interfaz muy sencilla	Free

Fuente: Los Autores

6.4 PLAN DE PRUEBAS

Para el desarrollo del análisis de seguridad informática sobre las aplicaciones móviles nativas, se debe diseñar, organizar e implementar una guía que contribuya a mantener el enfoque y objetivo del proceso, por esta razón, se ha creado un plan de pruebas basado en la Metodología del Proyecto de Seguridad Móvil OWASP Mobile Security Project, que establece un punto de partida para desarrolladores y equipos de seguridad sobre los requisitos necesarios para construir aplicaciones seguras, con el fin de reducir los riesgos de seguridad móvil, con el cual se pretender evaluar la seguridad de las aplicaciones móviles nativas seleccionadas enfocados en aspectos relacionados con los riesgos M2 almacenamiento de datos inseguro y M3 protección insuficiente en la capa de transporte, que permita la identificación de vulnerabilidades.

A- Recopilación de información sobre la Aplicación

Objetivo: reconocimiento de la aplicación para identificar la magnitud y alcance.

1- Nombre

2- Funcionalidad básica

3- ¿La aplicación realiza transacciones electrónicas?

☐ Si ☐ No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

☐ Si ☐ No

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

☐ NFC
☐ GPS
☐ Micrófono
☐ USB
☐ Bluetooth
☐ Cámara
☐ Sensores

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

☐ Telefonía (SMS, teléfono)
☐ Recepción de datos de aplicaciones y otros servicios en el dispositivo
☐ Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc.)
☐ Almacenamiento en la nube (Google Drive, Dropbox, iCloud)
☐ Contactos
☐ Google Wallet
☐ Correo electrónico

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

☐ Si ☐ No

7- Identificar las interfaces de red inalámbrica utilizadas:

☐ Wi-Fi (802.11) ☐ NFC ☐ Bluetooth

B- Análisis estático

Objetivo: identificación de vulnerabilidades de seguridad en el código fuente de la aplicación.

En esta fase se debe obtener el código fuente de la aplicación. Si no se tiene acceso al código fuente de la aplicación, se debe descompilar y/o desensamblar el binario de la misma.

General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

2- ¿La aplicación valida si el dispositivo esta rooteado?

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

C- Análisis dinámico

Objetivo: identificación de vulnerabilidades de seguridad de la aplicación durante el proceso de ejecución de la misma.

1- Instalar, configurar y utilizar la aplicación.

M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

5- Analizar almacenamiento de datos en cache.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

9- Determinar si se usan protocolos de comunicación de forma segura

6.5 RESULTADOS OBTENIDOS EN LA EJECUCIÓN DE PRUEBAS

Luego de instalar y configurar el ambiente de pruebas, se procede a desarrollar el plan de pruebas de seguridad sobre las aplicaciones móviles nativas seleccionadas utilizando las herramientas definidas previamente; los resultados obtenidos de este proceso se encuentran en el Anexo A- Ejecución de pruebas detallado por cada aplicación, allí se puede observar el resultado de cada una de los ítems evaluados para los riesgos OWASP Mobile M2 y M3.

6.6 ANÁLISIS DE RESULTADOS

Finalizado el proceso de ejecución del plan de pruebas sobre cada una de las aplicaciones, los resultados obtenidos se organizan, consolidan y analizan para diagnosticar el grado de seguridad de las aplicaciones móviles nativas evaluadas con relación a los riesgos OWASP Mobile M2 y M3.

El resultado de este análisis se agrupa en los ítems que se exponen a continuación

1- Permisos: se identificó que las aplicaciones solicitan para su ejecución permisos con nivel de protección “dangerous” (peligrosos) como se observa en la siguiente tabla.

Tabla 4. Análisis de permisos.

Permisos \ Aplicación	Wunderlist	Ted	SwiftKey + Emoji	Lumosity	Wish	Shazam	IFTTT	Groupon	Locket	Timehop
WRITE_EXTERNAL_STORAGE	X	X	X	X	X	X	X	X	X	X
READ_CONTACTS	X		X		X		X			
INTERNET		X	X	X	X	X	X	X	X	X
CAMERA					X		X	X	X	
READ_PHONE_STATE			X						X	
READ_SMS			X				X			X
ACCESS_COARSE_LOCATION						X		X	X	
ACCESS_FINE_LOCATION						X	X	X	X	
AUTHENTICATE_ACOUNTS						X		X		
NFC						X				
READ PHONE STATE						X	X	X		
RECORD_AUDIO						X				
READ_CALL_LOG							X			
RECEIVE_MMS							X			
RECEIVE_SMS							X			
SEND_SMS							X			
DISSABLE_KEYGUARD									X	
SYSTEM_ALERT_WINDOW									X	
MANAGE_ACCOUNTS								X		
USE_CREDENTIALS								X		

Fuente: los autores.

El permiso más usado es *write_external_storage* que permite a la aplicación leer, escribir y eliminar cualquier dato en el almacenamiento externo, convirtiéndose en una vulnerabilidad en la cual el atacante puede escribir u obtener datos sensibles del usuario.

En la aplicación Swift_key + Emoji sobresalen los permisos *read_sms* para leer mensajes guardados en la simcard del teléfono, *read_contacts* para el acceso a la información de contactos del dispositivo, los cuales no son necesarios para el funcionamiento de la aplicación convirtiéndose en una vulnerabilidad.

En la aplicación Wish sobresale el permiso de *camera* para tomar imágenes y videos, la cual no es necesaria para el funcionamiento de la misma.

En las aplicación Shazam, Locket y Groupon sobresalen los permisos de *access_coarse_location* y *access_fine_location* lo que le permite obtener la ubicación del usuario tomando como información la red de datos, el GPS y el Wi-fi, lo cual es una vulnerabilidad que esta aplicación pueda obtener información de Geolocalización que no se requiere para su normal funcionamiento. Adicionalmente Shazam y Groupon solicitan el permiso de *authenticate_acounts* que le permite actuar como autenticador de cuentas, lo cual puede ser potencialmente peligroso porque permite administrar las cuentas dentro del dispositivo.

En la aplicación IFTTT sobresalen los permisos *access_fine_location* para obtener la ubicación del usuario tomando como información la red de datos, el GPS y el Wi-fi, *read_call_log* para leer el registro de llamadas del dispositivo, *receive_mms* para monitorizar los mensajes MMS, *read_sms, receive_sms* y *send_sms* para el lectura, envío y recepción de mensaje SMS, lo que la hace una aplicación potencialmente vulnerable que puede ser usada por un atacante para el envío de mensajes de pago.

Timehop sobresalen el permiso *read_sms* para leer mensajes guardados en la simcard del teléfono que no es necesario para el funcionamiento de la aplicación.

2- Detección de root. se revisó en las aplicaciones la implementación de mecanismos de detección de root obteniendo como resultados los indicados en la siguiente tabla.

Tabla 5. Análisis de detección de root.

Aplicación	**Implementa mecanismo detección root**	
	Si	**No**
Wunderlist		X
Ted		X
SwiftKey + Emoji	X	
Lumosity		X
Wish	X	
Shazam	X	
IFTTT		X
Groupon		X
Locket		X
Timehop		X

Fuente: los autores.

Aplicaciones como Wunderlist, Ted, Lumosity no implementan ningún mecanismo de detección de root. Como consecuencia, la aplicación puede ejecutarse en un dispositivo rooteado, el cual no incluye muchas protecciones del sistema operativo que impida a los usuarios y programas acceso a la información sensible. Esto aumenta enormemente el riesgo de ataques como el robo de cuentas y recuperación de la información privada de aplicaciones de terceros.

3- Almacenamiento de datos inseguro (M2 Insecure Data Storage)

En este apartado se revisó el almacenamiento de información sensible en aspectos como bases de datos, almacenamiento externo, archivos (files), configuraciones (shared_prefs), cache, memoria y archivos de logs, los cuales se describen a continuación:

Tabla 6. Aplicaciones que almacenan datos sensibles de forma insegura

Aplicación	Base de Datos	Archivos	Configuración	Cache	Memoria	Log
Wunderlist	X	--	--	--	X	--
Ted	--	--	--	--	--	--
SwiftKey + Emoji	--	--	X	--	--	--
Lumosity	--	--	--	--	--	X
Wish	--	--	X	--	--	--
Shazam	--	--	X	--	--	--
IFTTT	--	--	X	--	--	--
Groupon	--	--	--	X	--	--
Locket	--	--	X	--	--	--
Timehop	--	--	--	X	--	--

Fuente: los autores

Los resultados obtenidos demuestran que el 90% de las aplicaciones móviles nativas evaluadas almacenan algún tipo de datos sensible del usuario o del dispositivo que pueden comprometer la seguridad de la información en caso que se presente un ataque malicioso.

A continuación se agrupan las aplicaciones según la información sensible expuesta:

IF by IFTTT almacena solamente información personal del usuario (nombres, dirección, ubicación, etc.).

Locket, Timehop, Swiftkey, Lumosity, Wish, Shazam y Groupon almacenan información de cuentas de usuario y/o correos electrónicos.

Wunderlist almacena información de identificación del dispositivo, nombre de usuario, correo electrónico y mensaje enviados.

Ted no almacena información sensible ya que su funcionalidad no utiliza un mecanismo de autenticación para su funcionamiento.

4- Protección Insuficiente en la capa de transporte (M3 Insufficient Transport Layer Protection)

En este apartado se realizó la captura y análisis del tráfico de red generado durante el uso de las aplicaciones móviles nativas con el objetivo de identificar la exposición de datos sensibles por parte de las mismas. A continuación se consolida los resultados obtenidos durante el proceso.

Tabla 7. Exposición de datos sensibles mediante tráfico de red

Aplicación	Protocolo HTTP	Protocolo HTTPS	Protocolo SSL
Wunderlist	X	X	X
Ted	X	X	--
SwiftKey + Emoji	--	X	--
Lumosity	--	X	X
Wish	--	X	X
Shazam	--	X	X
IFTTT	--	X	X
Groupon	--	X	X
Locket	X	--	--
Timehop	--	X	X

Fuente: los autores

Los resultados muestran un alto grado de vulnerabilidad de las aplicaciones en la seguridad de la capa de transporte ya que en la mayoría se identificó información del usuario, cuentas de correo, contraseñas, mensajes y en algunos casos información de tarjetas de crédito.

Aplicaciones como Groupon y Wish enfocadas a realizar compras de productos y/o servicios muestran un alto riesgo para la seguridad de los pagos ya que la información de tarjetas de crédito es transmitida sin aplicar mecanismos de cifrado.

A pesar que algunas aplicaciones implementan APIs de autenticación de terceros como Facebook o Gmail y usando protocolos seguros (HTTPS o SSL), no logran ofrecer una comunicación segura porque muchos de los parámetros que la aplicación transmite no se les aplica un mecanismo de cifrado seguro; en otros

casos la administración de sesión no se realiza correctamente lo cual permite ejecutar los *request* directamente en el navegador obteniendo información sensible desde el servidor.

A continuación se muestra el resumen de los datos sensibles que cada aplicación expone en la capa de transporte.

Tabla 8. Información sensible en la capa de transporte por aplicación

Aplicación / **Información Transmitida**	**Wunderlist**	**Ted**	**SwiftKey + Emoji**	**Lumosity**	**Wish**	**Shazam**	**IFTTT**	**Groupon**	**Locket**	**Timehop**
Datos personales										X
Contraseña	X	X		X	X		X	X		
Correo electrónico	X	X	X	X	X	X	X	X	X	X
IMEI								X	X	
Tarjeta de crédito					X			X		
Mensajes	X				X				X	X

Fuente: los autores

Una vez concluido el diagnóstico de seguridad informática de las aplicaciones móviles nativas evaluadas, se identifican las siguientes vulnerabilidades que pueden afectar la seguridad de los dispositivos móviles con sistema operativo Android versión Jelly Bean 4.1.2:

- Bases de datos con información confidencial sin cifrar, permitiendo leer el contenido y obtener información de cuentas de correo electrónico y/o contraseñas.

- Envío a través de la red de información confidencial en texto plano, permitiendo interceptar el tráfico de red y obtener la información de cuentas de correo electrónico, contraseñas, datos de tarjeta de crédito e incluso mensajes de texto enviados.

- La aceptación de los permisos solicitados por la aplicación permite el acceso a recursos propios del sistema operativo exponiendo información del usuario o el dispositivo (contactos, cuentas, registros de llamadas, lectura, envío y recepción de mensajes, datos de ubicación, fotos, correos, etc).

- El almacenamiento de información en medios externos (sd-card) no garantiza la seguridad de los datos debido a que estas unidades se

consideran públicas y pueden ser accedidas por otras aplicaciones exponiendo la información allí contenida.

- Falta de mecanismos para identificar rooteo del dispositivo móvil, permitiendo acceder a la información sensible de la aplicación o del sistema operativo aumentando el riesgo de ataques para el robo de cuentas y/o acceso a información privada de la aplicación.

7. PERSONAS QUE PARTICIPAN EN EL PROCESO

El desarrollo de la presente investigación requirió del siguiente personal:

Investigadores:

- Pedro Julio Colorado, Ingeniero de sistemas, estudiante de la especialización en seguridad informática de la UNAD, con experiencia en el diseño, desarrollo e implantación de software en el sector salud.

- Inírida Jeaneth Torres, Ingeniera de sistemas, estudiante de la especialización en seguridad informática de la UNAD, con experiencia en el manejo de sistemas de información de contratación estatal.

Director:

- Gabriel Mauricio Ramírez Villegas, Ingeniero de sistemas Magíster en Educación, Especialista en educación en línea, Especialista en desarrollo de aplicaciones móviles y seguridad móvil. Investigador del grupo TECNOGENESIS.

8. RECURSOS DISPONIBLES

Luego del análisis y revisión de la propuesta, para la ejecución del proyecto se requirió los siguientes recursos:

Físicos:

- Computadores: se requieren mínimo dos equipos de cómputo para la realización de pruebas de penetración, análisis, documentación de resultados y redacción del documento final de proyecto.
- Impresora: se requiere una impresora láser para la impresión de la documentación de la investigación.
- Smartphone: se requiere mínimo un dispositivo móvil con sistema operativo Android versión 4.1.2 Jelly Bean para realizar las pruebas.
- Servicios de Agua y Luz
- Servicio de Internet: el cual será utilizado para la búsqueda de información de apoyo, para la comunicación entre los investigadores y la comunicación con el asesor.
- Papelería e insumos para los equipos de cómputo e impresora: el cual será usado para la impresión de la documentación, análisis de la información.
- Transporte
- Imprevistos

Lógicos:

- Software emulador de Android para configuración de Android versión 4.1.2 Jelly Bean (API 16)
- Herramientas de software para realizar test de Penetración a dispositivos Smartphone.
- Distribución Linux Santoku.
- Suite de oficina (editor de texto, hoja de cálculo, presentación de diapositivas)

9. PRESUPUESTO

Para el desarrollo del proyecto se necesitó de recursos económicos para lograr con éxito su ejecución, los cuales se describen a continuación:

Tabla 9. Presupuesto de la investigación

Recurso	Egresos
A. Personal	
Investigador estudiante especialización $5.000.000 mes x 4.5 meses	$22.500.000
Investigador estudiante especialización $5.000.000 mes x 4.5 meses	$22.500.000
B. Software	
Licenciamiento Suite de oficina (editor de texto, hoja de cálculo, presentación de diapositivas) x 2 equipos de computo	$300.000
Adquisición de software emulador para configuración de Android versión 4.1.2 Jelly Bean	$500.000
Adquisición de herramientas de software para realizar test de Penetración a dispositivos Smartphone.	$500.000
Adquisición de herramientas de software para evaluación de aplicaciones móviles	$500.000
B. Equipos	
Adquisición de dos (2) equipos de cómputo	$4.000.000
Adquisición de una (1) Impresora láser	$400.000
Adquisición Smartphone Android versión 4.1.2 Jelly Bean	$600.000
C. Servicios	
Internet x 4.5 meses	$315.000
Agua x 4.5 meses	$135.000
Energía x 4.5 meses	$270.000
D. Viajes	
Transporte x 4.5 meses	$1.000.000
E. Materiales	
Papelería e insumos para los equipos de cómputo e impresora x 4.5 meses	$600.000
F. Imprevistos	
Imprevistos	$800.000
Total	$54.920.000

Fuente: Los Autores

10. CRONOGRAMA

Para el desarrollo del proyecto se determinó el siguiente diagrama GANTT, basados en una dedicación semanal de 20 horas

Tabla 10. Cronograma del proyecto

Actividades \ Semana	Mes 1				Mes 2				Mes 3				Mes 4				Mes 5	
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2
Revisión y documentación de la arquitectura y modelo de seguridad del sistema operativo Android.																		
Identificación vulnerabilidades y amenazas de seguridad en dispositivos Smartphone con sistema operativo Android a nivel de aplicaciones																		
Realización de pruebas de hacking ético al dispositivo móvil Smartphone con sistema operativo Android Jelly Bean versión 4.1.2.																		
Construcción del informe del proceso de evaluación de seguridad de las aplicaciones móviles nativas analizadas de acuerdo a los resultados obtenidos																		
Construcción del documento final del proyecto de investigación																		
Presentación de documento final de la investigación																		
Sustentación del proyecto																		

Fuente: Los Autores

11. CONCLUSIONES

Finalizado el análisis de los resultados de las pruebas de seguridad ejecutadas se puede concluir que el objeto de estudio de la presente investigación ha sido cumplido, puesto que se encontraron suficientes evidencias que demuestran las vulnerabilidades de las aplicaciones móviles nativas frente a los riesgos M2 y M3 de OWASP Mobile.

A nivel de aplicaciones móviles para sistemas operativo Android no es suficiente el distintivo “mejores aplicaciones de 2014” otorgado en Google Play Store, dado que este se asigna teniendo en cuenta criterios superficiales como el diseño, el número de descargas y la popularidad entre los usuarios, omitiendo ítems tan importantes como el grado de seguridad frente a posibles amenazas informáticas y la confidencialidad de la información sensible.

El componente de permisos del modelo de seguridad del sistema operativo Android representa un alto riesgo de seguridad porque los usuarios no tienen conocimiento claro del riesgo e impacto de los permisos que conceden a una aplicación en el proceso de instalación, y como se evidenció en el desarrollo del proyecto, una aplicación con un permiso inadecuado puede comprometer toda la seguridad del dispositivo móvil.

El uso de mecanismos de autenticación (APIs) de terceros reconocidos en el mercado no siempre garantiza la confidencialidad de la información pues estos deben ser implementados pensando en primer lugar en la seguridad de los datos y posteriormente en la integración de los componentes o aplicaciones.

Los desarrolladores de aplicaciones móviles para sistema el operativo Android deberían implementar el uso de metodologías, modelos y estándares orientados a brindar un mayor de grado de seguridad informática a los usuarios de la plataforma; OWASP Mobile Security Project es un claro ejemplo de ello, dado que en este proyecto contribuyó para definir los aspectos más representativos donde identificar posibles vulnerabilidades de seguridad en las aplicaciones analizadas.

Las aplicaciones móviles nativas analizadas en el presente estudio constituyen un pequeño porcentaje frente a la cantidad de aplicaciones disponibles en el Google Play Store para Android, las cuales son descargadas e instaladas a diario por los usuarios sin conocer los riesgos a los que se exponen, incrementando así las estadísticas de ataques de seguridad o proliferando la distribución de malware, virus o troyanos entre la comunidad.

La seguridad de la información debe constituirse como un elemento clave en un ecosistema móvil, los profesionales en seguridad informática afrontan el reto de generar conciencia en todos los componentes (usuarios, desarrolladores y

proveedores) para que definan y establezcan políticas, procedimientos, guías y técnicas que ayuden a reducir las vulnerabilidades que se puedan presentar en las aplicaciones móviles.

Los resultados obtenidos en la presente investigación abren la puerta para que la comunidad académica de la UNAD continúe el desarrollo de proyectos futuros que aporten a mejorar la seguridad en los dispositivos móviles, se proponen temas como: seguridad de aplicaciones en otros sistemas operativos móviles como iOS o Windows Phone, el desarrollo de herramientas de software para automatizar pruebas de seguridad, para identificar malware en aplicaciones móviles, entre otros.

12. BIBLIOGRAFÍA

AGUILERA, Purificación. Introducción a la Seguridad Informática. Editorial Editex. ISBN 978-84-9003-106-3. p. 9.

ANDROID DEVELOPERS [en línea] [citado el 18 octubre, 2014]. Disponible en internet: http://developer.android.com/about/index.html

ANDROID INC. Permission. [en línea] [citado el 16 mayo, 2014]. Disponible en internet: https://developer.android.com/guide/topics/manifest/permission-element.html#plevel

ANDROID. Android Security Overview. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: http://source.android.com/devices/tech/security/index.html

ANDROID. The Android Source Code. Governance Philosophy. [en línea] [citado el 18 octubre, 2014]. Disponible en internet: http://source.android.com/source/index.html#governance-philosophy

ANDROID. The Android Story. [en línea] [citado el 18 octubre, 2014] Disponible en internet http://www.android.com/history/

APPLE INC. Apple presenta el nuevo iPhone 4. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: http://www.apple.com/la/pr/library/2010/06/07Apple-Presents-iPhone-4.html

ARXAN. TECHNOLOGIES. Security for Android Java Mobile Applications. [en línea] [citado el 12 Enero, 2015]. Disponible en internet: https://www.arxan.com/products/mobile/guardit-for-java/

ASOCIACIÓN COLOMBIANA DE INGENIEROS DE SISTEMAS Y MUSEO COLOMBIANO DE INFORMÁTICA, E IBM. Historia de la Computación Historia de la Computación. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet: http://www.acis.org.co/archivosAcis/HistoriadelaComputacion.pdf

BAZ, Arturo, et al. Dispositivos móviles. Universidad de Oviedo. [en línea] [citado el 25 Mayo, 2014]. Disponible en internet: http://156.35.151.9/~smi/5tm/09trabajos-sistemas/1/Memoria.pdf. p. 23.

BERGMAN, Neil et al. Hacking Exposed Mobile. Security secrets & solutions. Editorial Mc Graw Hill. 2013. ISBN: 978-0-07-181702-8

BLACKBERRY LIMITED. What is the BlackBerry Family?. [en línea] [citado el 18 octubre, 2014] Disponible en internet: http://blogs.blackberry.com/2014/09/what-is-the-blackberry-family/

CALDER, Alan y WATKINS, Steve. *IT Governance: A Manager's Guide to Data Security& BS 7799/ISO 17799*. Citado por SOLARTE, Francisco y BENAVIDES, Miriam. Riesgos y Control Informático. [en línea]. [citado el 29 de octubre, 2014]. Disponible en internet:http://datateca.unad.edu.co/contenidos/233004/riesgos/leccin_3_anlisis_de_riesgos.html

CANDELA, Santiago, et al. Fundamentos de sistemas operativos: teoría y ejercicios resueltos. [en línea]. [citado el 16 de mayo, 2014]. Disponible en internet http://books.google.es/books?id=fRK3lbTrNy4C&dq=sistemas+operativos&source=

CONCIL OF EUROPE. Convenio sobre la Ciberdelincuencia. Budapest, 23.XI.2001. [en línea] [citado el 15 diciembre, 2014]. Disponible en internet: http://www.coe.int/t/dghl/cooperation/economiccrime/Source/Cybercrime/TCY/ETS_185_spanish.PDF

CONSEJO NACIONAL CONSULTIVO DE CYBER-SEGURIDAD. Malware en Smartphones. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: http://www.bdigital.org/Documents/Malware_Smartphones.pdf. p. 4.

CVE DETAILS. Android: Vulnerability Statistics. [en línea] [citado el 8 diciembre, 2014]. Disponible en internet: http://www.cvedetails.com/product/19997/Google-Android.html?vendor_id=1224

DOMINGO, Marc. Seguridad en dispositivos móviles. Universidad Oberta de Catalunya. CC-BY-SA-PID_00178751.

DWIVEDI, Himanshu. Mobile Aplication Security. Citado por RAMIREZ, Gabriel. La seguridad en aplicaciones móviles: estrategias en el mundo actual. [en línea]. [citado el 18 de octubre, 2014]. Disponible en internet: http://datateca.unad.edu.co/contenidos/233016/Articulos/La_Seguridad_en_Aplicaciones_Moviles_Estrategias_en_el_Mundo_Actual_Gabriel_Ramirez.pdf. p. 3.

ENCK, William, et al. A study of Android Application Security. [en línea] [citado el 12 noviembre, 2014]. Disponible en internet: http://www.cs.rice.edu/~sc40/pubs/enck-sec11.pdf

ENRIQUEZ, Juan y CASAS, Sandra. Usabilidad en aplicaciones móviles. Informes Científicos y Técnicos. Publicaciones de actualización continua. Universidad

Nacional de la Patagonia Austral [en línea] [citado el 18 octubre, 2014] Disponible en internet http://ict.unpa.edu.ar/files/ICT-UNPA-62-2013.pdf p. 11.

ESET LATINOAMÉRICA. Tendencias 2014: El desafío de la privacidad en Internet. 2014. [en línea] [citado el 28 Mayo, 2014]. Disponible en internet: http://www.eset-la.com/pdf/tendencias_2014_el_desafio_de_la_privacidad_en_internet.pdf

ESET LATINOAMÉRICA. Top 10 de OWASP de vulnerabilidades en aplicaciones móviles. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: http://www.welivesecurity.com/la-es/2014/02/26/top-10-owasp-vulnerabilidades-aplicaciones-moviles/

FIGUEREDO, Oscar. Sistemas Operativos para Dispositivos Móviles. Entérese, 74-78. 2006, citado por POLANCO, Kristel y BEAUPERTHUY, José. “Android” el sistema operativo de Google para dispositivos móviles, Revista Científica Electrónica Ciencias Gerenciales. [en línea]. [citado el 16 de mayo, 2014]. Disponible en internet: http://www.revistanegotium.org.ve/pdf/19/art4.pdf p. 81

FLING, Brian. Mobile Design and Development. Capitulo 6. Editorial O’Relly. ISBN 978-0-596-15544-5.

GARTNER. Gartner Says Smartphone Sales Grew 46.5 Percent in Second Quarter of 2013 and Exceeded Feature Phone Sales for First Time. [en línea] [citado el 16 mayo, 2014] Disponible en internet: http://www.gartner.com/newsroom/id/2573415

GARZÓN, Juan. La evolución de los celulares Samsung Galaxy S: tercera parte. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: http://www.cnet.com/es/noticias/samsung-galaxy-s5-evolucion/ gbs_navlinks_s p.4.

GIRONES, Jesus. El gran libro de Android. Ediciones Marcombo. Tercera Edición. 2013. ISBN: 978-84-267-2078-8.

GONZÁLEZ, Yina y CASTAÑO, Wilson. Fundamentos de Seguridad de la Información. Universidad Nacional Abierta y a Distancia. 2012. p. 37.

GOOGLE INC. Condiciones del Servicio de Google. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: https://www.google.com.co/intl/es-419/policies/terms/regional.html

GOOGLE INC. Condiciones del Servicio de Google. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: https://play.google.com/intl/ALL_es/about/developer-distribution-agreement.html

GRANADOS, Gerardo. Sistemas Distribuidos. Universidad Nacional Abierta y a Distancia. [en línea] [citado el 18 Octubre, 2014]. Disponible en internet: http://datateca.unad.edu.co/contenidos/208017/ContLin/leccin_3_comunicacin_en_los_sistemas_distribuidos.html.

GUNASEKERA, Sheran. Android Apps Security. Editorial Apress. 2012. p. 32. ISBN 978-1-4302-4063-1.

HILL, Simón. A history of Samsung's Galaxy pones and tablets, from the S1 to the S4. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: http://www.digitaltrends.com/mobile/history-of-samsungs-galaxy-phones-and-tablets/

HIMANSHU, Dwivedi. Mobile Application Security. Editorial Mc Graw Hill. P. 22 ISBN 978-0-07-163356-7.

IBM CORPORATION. Cómo garantizar la seguridad de las aplicaciones para dispositivos móviles. Software Group, 2012. p. 3-4.

ICSA LABS. Will the Promise of Hybrid Mobile Apps Outweigh New Security Concerns?. [en línea] [citado el 6 diciembre, 2014]. Disponible en internet: https://www.icsalabs.com/blogs/will-promise-hybrid-mobile-apps-outweigh-new-security-concerns

INTERNATIONAL DATA CORPORATION IDC. Android and iOS Continue to Dominate the Worldwide Smartphone Market with Android Shipments Just Shy of 800 Million in 2013. 2014. [en línea] [citado el 20 Mayo, 2014]. Disponible en internet http://www.idc.com/getdoc.jsp?containerId=prUS24676414

iOS 8. El sistema operativo móvil más avanzado del mundo. Y de qué manera. [en línea] [citado el 18 octubre, 2014] Disponible en internet https://www.apple.com/es/ios/what-is/

IPHONE WORLD. Apple quiere que todos los dispositivos integren Siri. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: http://www.iphoneworld.com.es/2012/01/apple-quiere-que-todos-los-dispositivos.html

KASPERSKY LAB. Abril de 2014. ¿Por qué quieren los cibercriminales tu Smartphone?. [en línea] [citado el 25 Mayo, 2014]. Disponible en internet: http://newsroom.kaspersky.eu/fileadmin/user_upload/es/Downloads/Kaspersky_pressrelease_Por_qu%C3%A9_quieren_los_cibercriminales_tu_smartphone.doc.pdf p. 1.

KASPERSKY LAB. Mobile Malware Evolution: 2013. [en línea] [citado el 16 mayo, 2014]. Disponible en internet: http://blog.kaspersky.com/mobile-malware-evolution-2013/

KASPERSKY LAB. *Kaspersky Security Bulletin 2013. Overall statistics for 2013.* Diciembre de 2013. [en línea] [citado el 25 Mayo, 2014]. Disponible en internet: https://www.securelist.com/en/analysis/204792318/Kaspersky_Security_Bulletin_2013_Overall_statistics_for_2013

KUMAR, Ajitesh. Java - Four Security Vulnerabilities Related Coding Practices to Avoid [en línea] [citado el 12 Enero, 2015]. Disponible en internet: http://java.dzone.com/articles/java-four-security

MARRUGO, Iván. Colombia y la cooperación internacional en los delitos informáticos. [en línea] [citado el 15 diciembre, 2014]. Disponible en internet: http://www.ambitojuridico.com/BancoConocimiento/N/noti-140130-06colombia_y_la_cooperacion_internacional_en_los_delitos_inform/noti-140130-06colombia_y_la_cooperacion_internacional_en_los_delitos_inform.asp?IDObjetoSE=17572

MICROSOFT CORPORATION. Windows Phone 8 update history. [en línea] [citado el 18 octubre, 2014] Disponible en internet: http://www.windowsphone.com/en-us/how-to/wp8/basics/windows-phone-8-update-history

MIERES, Jorge. Ataques informáticos. Debilidades de Seguridad comúnmente explotadas. [en línea]. [citado el 30 de noviembre, 2014]. Disponible en internet: https://www.evilfingers.com/publications/white_AR/01_Ataques_informaticos.pdf

MINISTERIO DE TECNOLOGÍAS Y LA INFORMACIÓN. Aplicaciones móviles colombianas con calidad de exportación [en línea] [citado el 2 marzo, 2015]. Disponible en internet: <http://www.mintic.gov.co/portal/604/w3-article-8396.html>

MOHINI, Tiwari, et al. Review on Android and Smartphone Security. [en línea] [citado el 8 diciembre, 2014]. Disponible en internet: http://www.academia.edu/6375839/Review_on_Android_and_Smartphone_Security

MONTOYA, Juan. El Smartphone innovado. [en línea]. [citado el 10 de octubre, 2014]. Disponible en internet: https://sites.google.com/site/elsmartphoneinnovando/historia-del-smartphone

MOZILLA. Faq de Firefox OS. Acerca de Firefox Os. [en línea] [citado el 18 octubre, 2014] Disponible en internet: https://www.mozilla.org/es-ES/firefox/os/faq/

NATIONAL TECHNOLOGY ASSISTANCE PROJECT. II. Understanding the Mobile Ecosystem. [en línea]. [citado el 17 de noviembre, 2014]. Disponible en internet: http://lsntap.org/book/export/html/3555

NOWSECURE. Secure Mobile Development Best Practices. [en línea] [citado el 9 diciembre, 2014]. Disponible en internet: https://www.nowsecure.com/resources/downloads/secure-mobile-development/

OPEN SIGNAL. Android Fragmentation visualized. [en línea]. [citado el 27 de noviembre, 2014]. Disponible en internet: http://opensignal.com/reports/2014/android-fragmentation/

OWASP Open Web Application Security Project. Category:Vulnerability. [en línea]. [citado el 23 de octubre, 2014]. Disponible en internet: https://www.owasp.org/index.php/Category:Vulnerability

OWASP. OWASP Mobile Security Project. [en línea] [citado el 30 octubre, 2014]. Disponible en internet: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project/

RAMÍREZ, Gabriel. La importancia de la computación móvil: pasado, presente y futuro. Revista Especializada en Telecomunicaciones, Electrónica y Sistemas. Universidad Nacional Abierta y a Distancia. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet: http://datateca.unad.edu.co/contenidos/201493/Articulos/articulo_6-6.pdf. Volumen 2, Número 2. p. 3.

RAMÍREZ, Gabriel. Seguridad en aplicaciones móviles. Contenido didáctico. Universidad Nacional Abierta y a Distancia. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet: http://datateca.unad.edu.co/contenidos/201493/CONTENIDO%20DIDACTICO%20EXE1/leccion_3_evolucion_de_la_computacion_movil.html.

RAMIREZ, Gabriel. Seguridad en aplicaciones móviles. Universidad Nacional Abierta y a Distancia, Palmira, 2013. p. 126.

SAMBASIVAN, D., et al., Generic Framework for Mobile Application Development. The Second Asian Himalayas International Conference on Internet. Citado por ENRIQUEZ, Juan y CASAS, Sandra. Usabilidad en aplicaciones móviles. [en línea]. [citado el 18 de octubre, 2014]. Disponible en internet: http://ict.unpa.edu.ar/files/ICT-UNPA-62-2013.pdf. p. 11-12.

SAMSUNG. Samsung revela Galaxy S5, un Smartphone pensado en las necesidades de los consumidores. [en línea]. [citado el 10 de octubre, 2014].

Disponible en internet: http://www.samsung.com/co/news/global/samsung-introduces-galaxy-s5

SEACORD, Robert. CERT® C Coding Standard, Second Edition, The: 98 Rules for Developing Safe, Reliable, and Secure Systems. [en línea] [citado el 12 Enero, 2015]. Editorial Adisson Wesley Disponible en internet: https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=146440541

SENADO DE LA REPUBLICA. Ley 1273 de 2009. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: http://www.secretariasenado.gov.co/senado/basedoc/ley_1273_2009.html

SENADO DE LA REPUBLICA. Ley 599 de 2000. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: http://www.secretariasenado.gov.co/senado/basedoc/ley_0599_2000.html

SIX, Jeff. Application.Security for the Android Platform. Editores O'Reilly. 2012. p. 15. ISBN. 978-1-449-31507-8

SOPHOS. Sophos Mobile Security Threat Report. Launched at Mobile World Congress, 2014. [en línea] [citado el 16 mayo, 2014]. Disponible en internet: http://www.sophos.com/en-us/medialibrary/PDFs/other/sophos-mobile-security-threat-report.pdf

STANLEY Morgan. The mobile Internet Report. [en línea] [citado el 18 octubre, 2014] Disponible en internet: http://www.morganstanley.com/institutional/techresearch/pdfs/2SETUP_12142009_RI.pdf

STROUSTRUP, Bjarne. An Overview of the C++ Programming Language. The Handbook of Object Technology (Editor: Saba Zamir). CRC Press LLC, Boca Raton. 1999. ISBN 0-8493-3135-8. Disponible en internet: http://www.stroustrup.com/crc.pdf

TALUKDER, Asoke y YAVAGAL, Roopa. Mobile computing. Technology, Applications and Service creation. Editorial Tata McGraw Hill. 2005. ISBN-13: 978-0-07-058807-3.

TECHTARGET. Android fragmentation: More OS versions, more problems. [en línea]. [citado el 5 de Diciembre, 2014]. Disponible en: http://searchconsumerization.techtarget.com/feature/Android-fragmentation-More-OS-versions-more-problems

THE NEXT WOMEN BUSINESS MAGAZIN. The Mobile Ecosystem: How Can it Benefit Your Business?. [en línea]. [citado el 17 de noviembre, 2014]. Disponible

en internet: http://www.thenextwomen.com/2013/01/09/mobile-ecosystem-how-can-it-benefit-your-business

TIZEN. About. [en línea] [citado el 18 octubre, 2014]Disponible en internet: https://www.tizen.org/about

U.S. GOVERNMENT PRINTING OFFICE. §1029. Fraud and related activity in connection with access devices. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: http://www.gpo.gov/fdsys/pkg/USCODE-2010-title18/html/USCODE-2010-title18-partI-chap47-sec1029.htm

U.S. GOVERNMENT PRINTING OFFICE. §1030. Fraud and related activity in connection with computers. [en línea] [citado el 10 noviembre, 2014]. Disponible en internet: http://www.gpo.gov/fdsys/pkg/USCODE-2010-title18/html/USCODE-2010-title18-partI-chap47-sec1030.htm

UNIVERSIDAD CARLOS III DE MADRID. Programación en dispositivos móviles portables. Arquitectura Android [en línea]. [citado el 28 Mayo, 2014]. Disponible en internet: https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android

UNIVERSIDAD NACIONAL DE LUJAN. Departamento de Seguridad Informática. Análisis a la seguridad de la información. [en línea]. [citado el 29 de octubre, 2014]. Disponible en internet: http://www.seguridadinformatica.unlu.edu.ar/?q=node/12

VIAFORENSICS. 42+ Best practices: Secure mobile development for iOS and Android. 2012. [en línea] [citado el 8 octubre, 2014]. Disponible en internet: https://viaforensics.com/resources/reports/best-practices-ios-android-secure-mobile-development/mobile-security-primer/

VISIONMOBILE. Report Mobile Megatrends 2012. Citado por RAMÍREZ, Gabriel. La seguridad en aplicaciones móviles: estrategias en el mundo actual, Artículo. [en línea]. [citado el 18 de octubre, 2014]. Disponible en internet: http://datateca.unad.edu.co/contenidos/233016/Articulos/La_Seguridad_en_Aplicaciones_Moviles_Estrategias_en_el_Mundo_Actual_Gabriel_Ramirez.pdf. p. 1.

WEISER, Mark. [en línea] [citado el 15 Octubre, 2014]. Disponible en internet: http://www.ubiq.com/hypertext/

ZHAUNIAROVICH, Yury. AndroidTM Security (and Not) Internals. 2014. License Creative Commons. p. 7.

ZHOU, Yajin y XUXIAN Jiang. Android Malware Genome Project. [en línea] [citado el 8 diciembre, 2014]. http://www.malgenomeproject.org/

Anexo A. Resultados ejecución de Pruebas

I. Aplicación Wunderlist

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
Wunderlist (com.wunderkinder.wunderlistandroid)

2- Funcionalidad básica
Aplicación que permite crear listas de tareas y sincronizarlas entre diferentes dispositivos y compartir listas con otros usuarios. Tiene como características: trabajo colaborativo, creación listas públicas, creación listas inteligentes, sincronización de la aplicación con Facebook y otras redes.

3- ¿La aplicación realiza transacciones electrónicas?

[X] Si [ ] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[X] Si [ ] No

Figura 30. Wunderlist Permisos

Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

[ ]	NFC	[ ]	Bluetooth
[ ]	GPS	[X]	Cámara
[ ]	Micrófono	[ ]	Sensores
[X]	USB		

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

[ ]	Telefonía (SMS, teléfono)	[X]	Contactos
[ ]	Recepción de datos de aplicaciones y otros servicios en el dispositivo	[ ]	Google Wallet
[X]	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)	[ ]	Correo electrónico
[ ]	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 31. Wunderlist interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[X] Si [ ] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[X] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

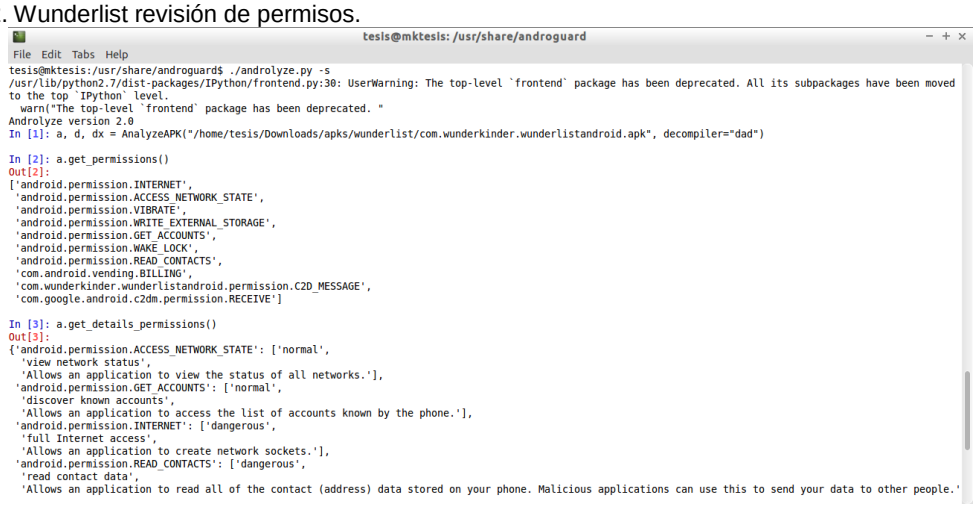
B- Análisis estático

General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.
El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

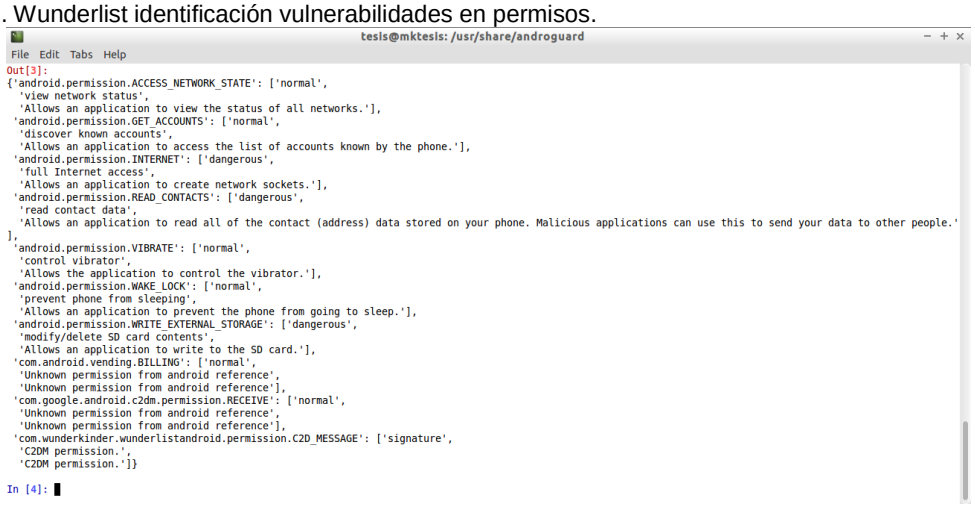
- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.
- READ_CONTACTS permite el acceso a la información de contactos del dispositivo, sin poder controlar que destino se le dará a estos datos.

Figura 32. Wunderlist revisión de permisos.



Fuente: Los Autores.

Figura 33. Wunderlist identificación vulnerabilidades en permisos.



Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?
No. En la revisión del código fuente no se encontró uso de métodos de validación de este parámetro en la búsqueda de instrucciones con los comandos “xbin”, “su”, “sbin”, “system”

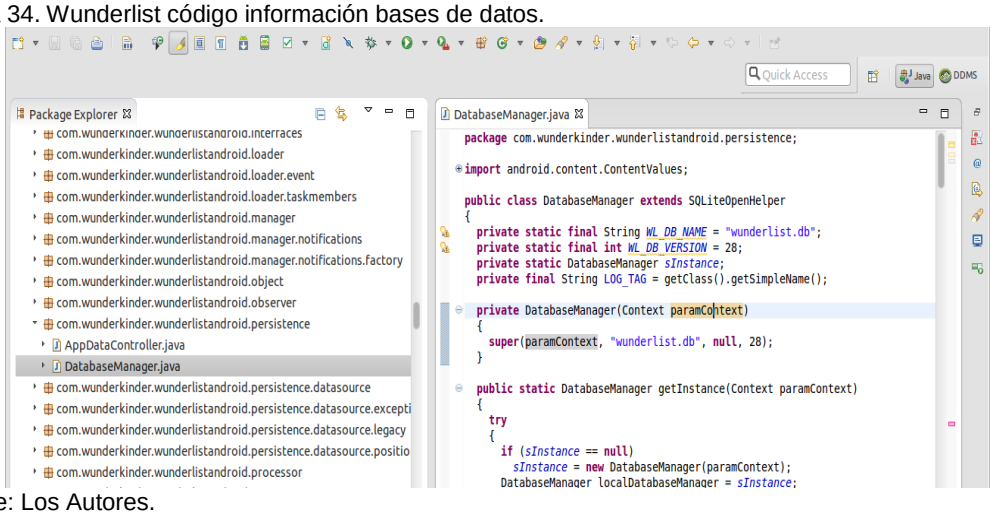
Los dispositivos rooteados no incluyen todas las protecciones de seguridad en el sistema operativo permitiendo el acceso total a información y datos de aplicaciones.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa una base de datos llamado *wunderlist.db*, el archivo que muestra esta información es: *com/wunderkinder/wunderlistandroid/persistence/DatabaseManager.java*

Figura 34. Wunderlist código información bases de datos.



Fuente: Los Autores.

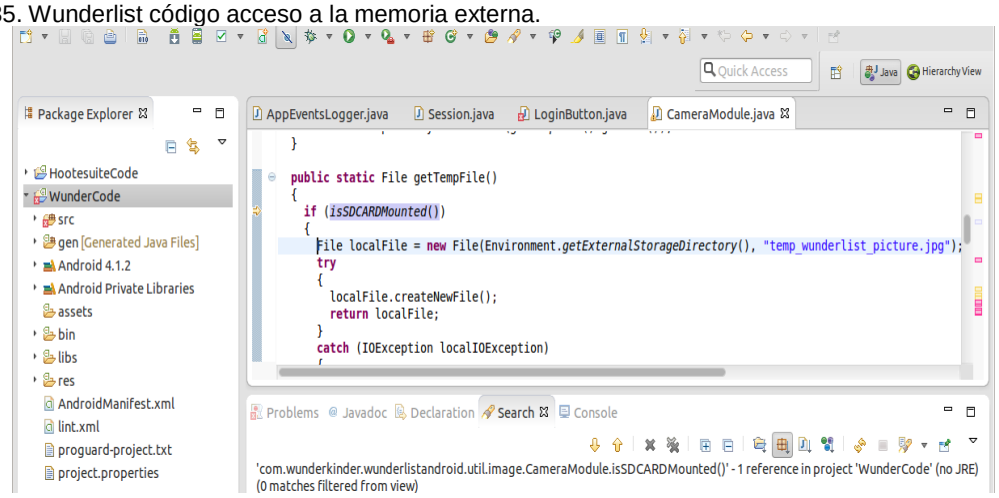
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

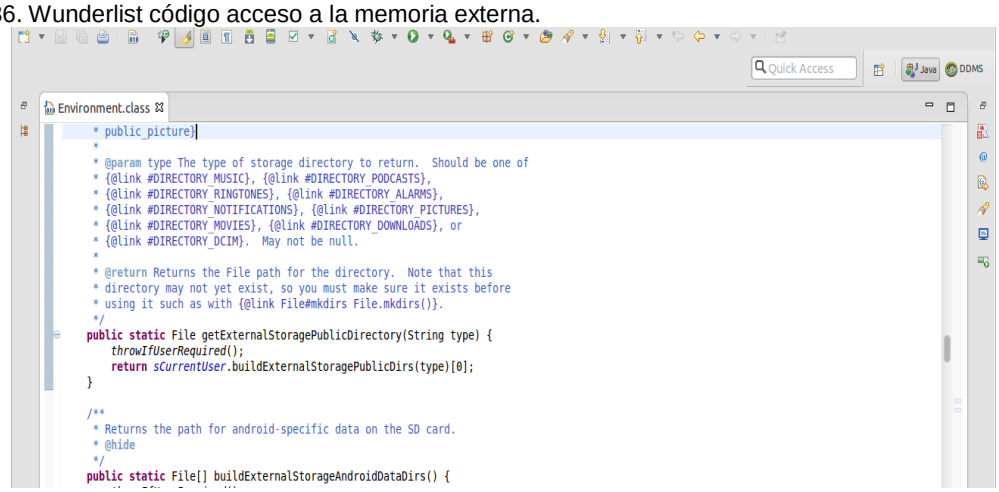
En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 35. Wunderlist código acceso a la memoria externa.



Fuente: Los Autores.

Figura 36. Wunderlist código acceso a la memoria externa.

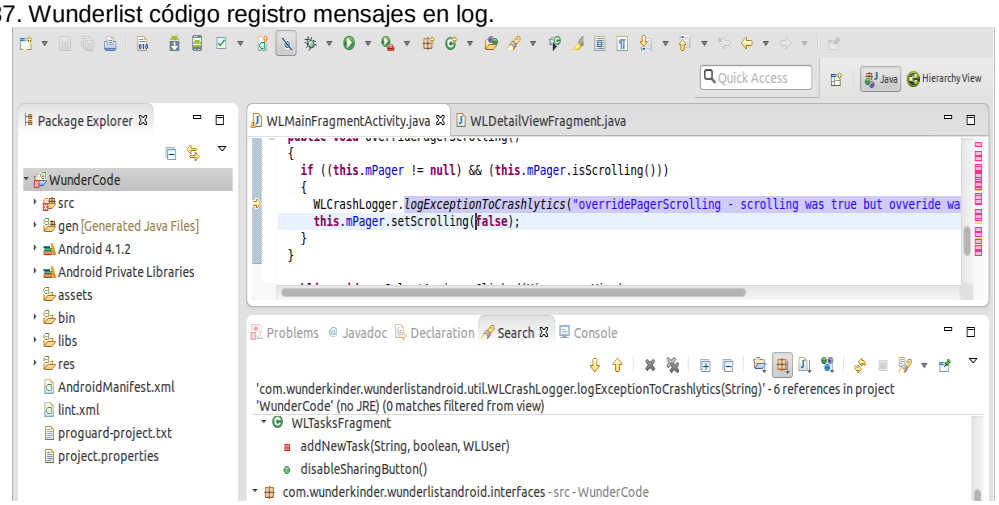


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 37. Wunderlist código registro mensajes en log.



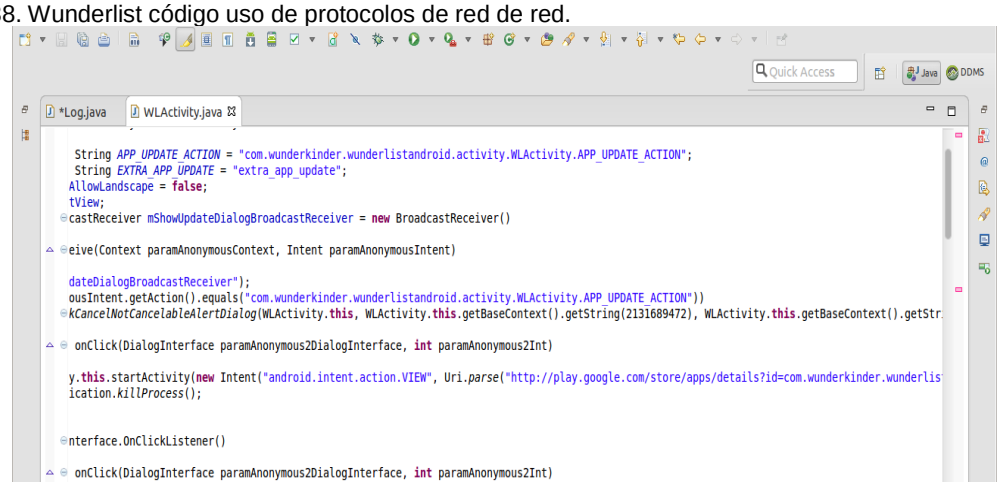
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 38. Wunderlist código uso de protocolos de red de red.

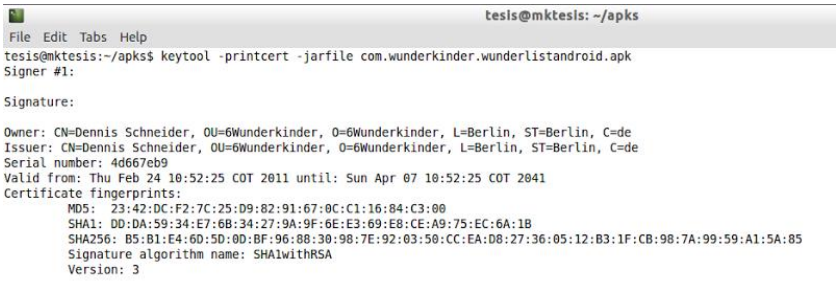


Fuente: Los Autores.

7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

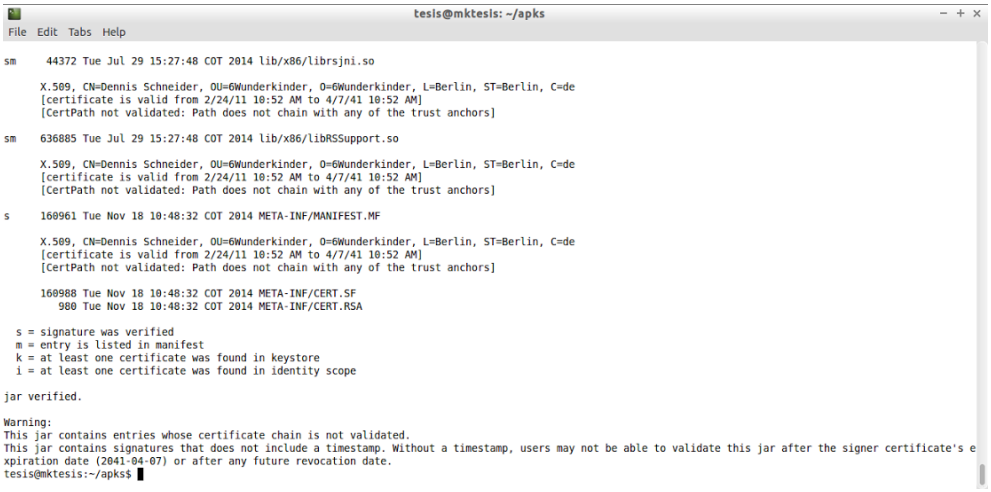
Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 39. Wunderlist información del certificado.



Fuente: Los Autores.

Figura 40. Wunderlist verificación del certificado.



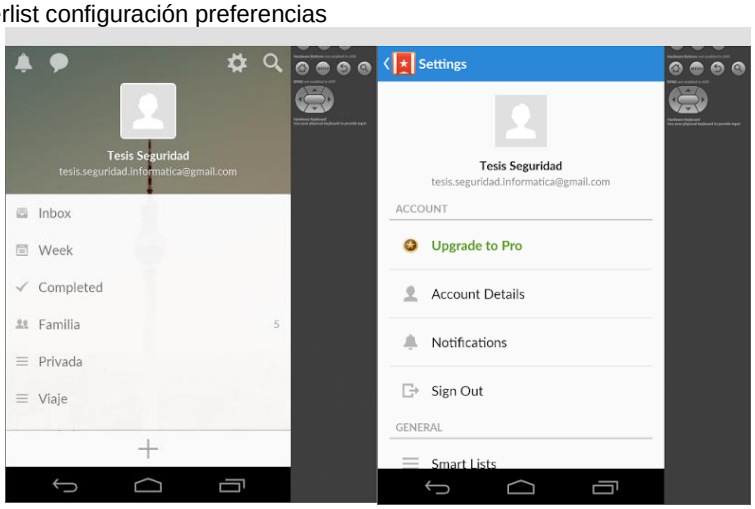
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 41. Wunderlist configuración preferencias



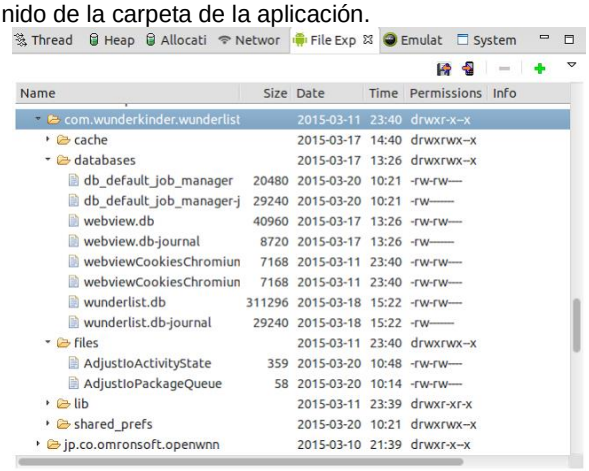
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio "/data/data" crea las carpetas denominada "com.wunderwinder.wunerlistandroid" con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

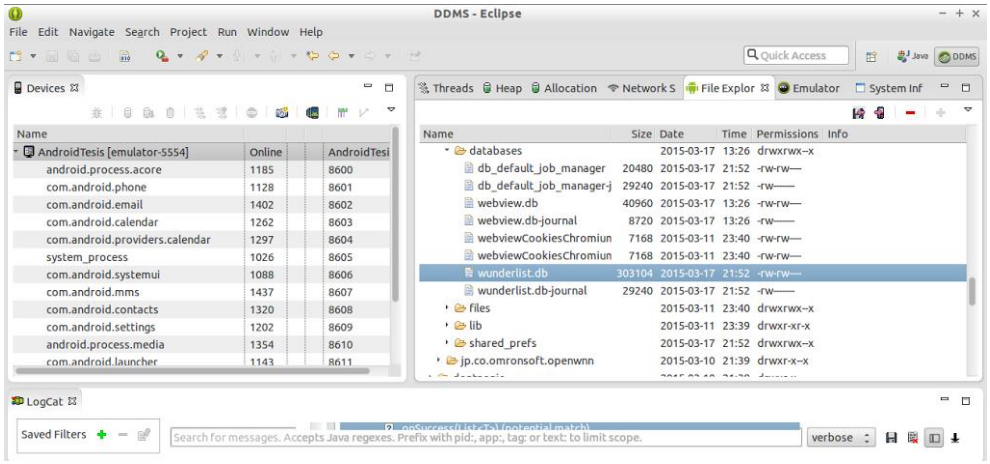
Figura 42. Wunderlist contenido de la carpeta de la aplicación.



Fuente: Los Autores.

Adicionalmente se observa en la carpeta "/sdcard" la creación de una subcarpeta WunderListFiles para el almacenamiento de archivos, la cual se encuentra vacía.

Figura 43. Wunderlist ubicación archivo base de datos



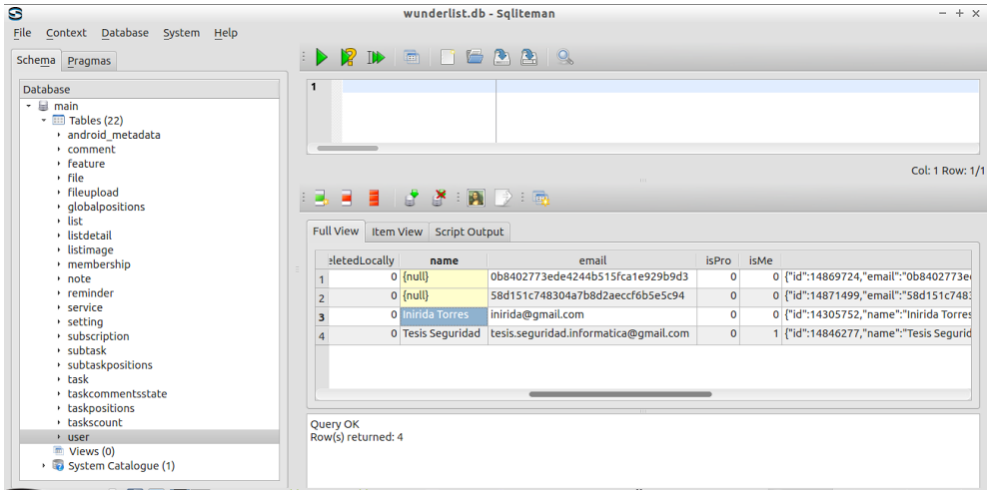
Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “webview.db” y “wunderlist.db”.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “wunderlist.db” se observan varias tablas entre las cuales se encuentra la tabla “user” la cual una vez examinada se observa en su contenido los datos sensibles de usuario (id, nombre, correo electrónico) no se encuentran cifrados y la tabla “comment” que muestra el contenido de los mensajes enviados, lo cual se convierte en una vulnerabilidad que viola el principio de confidencialidad de la información.

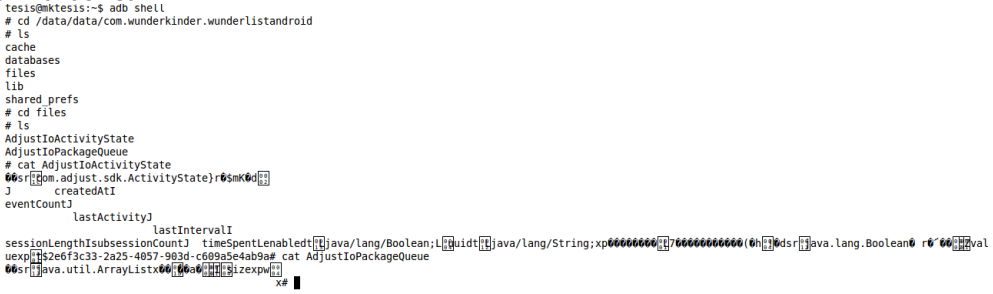
Figura 44. Wunderlist contenido de la tabla user.



Fuente: Los Autores.

En la carpeta files se encuentran dos archivos, que no manejan información sensible. No se observan archivos xml.

Figura 45. Wunderlist

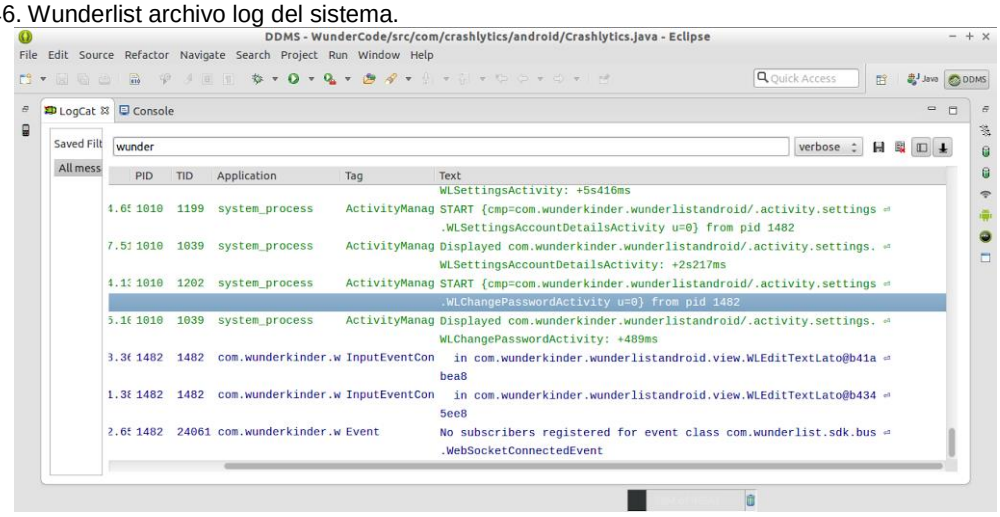


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log no se observa el almacenamiento de información sensible, solo muestra información de las actividades generadas.

Figura 46. Wunderlist archivo log del sistema.



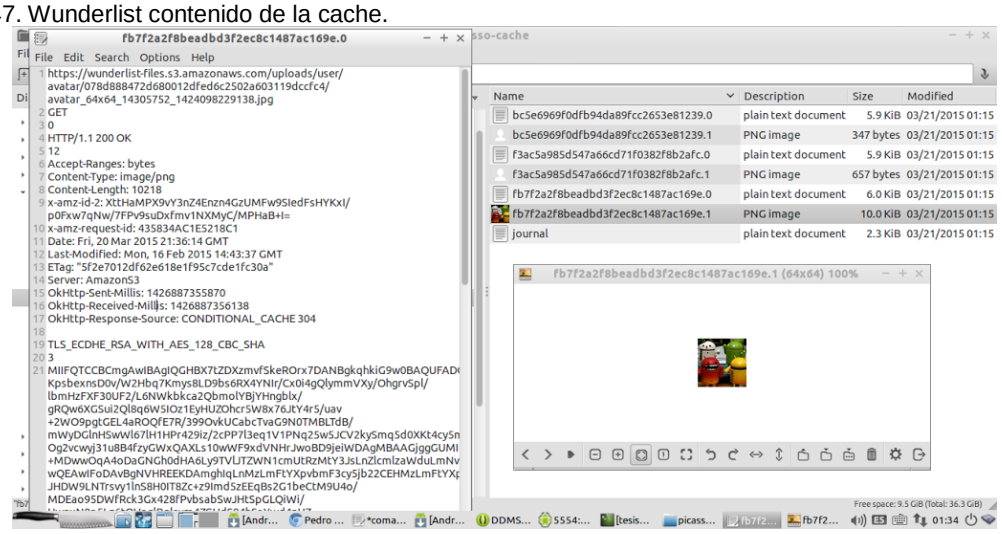
Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación en donde se encontraron varias subcarpetas entre las cuales llama la atención la carpeta “picasso-cache” en la cual se encuentran las fotos de los usuarios con los

cuales se ha compartido listas y archivo de texto plano que contiene la información de la foto el cual parte de su contenido se encuentra cifrado.

Figura 47. Wunderlist contenido de la cache.

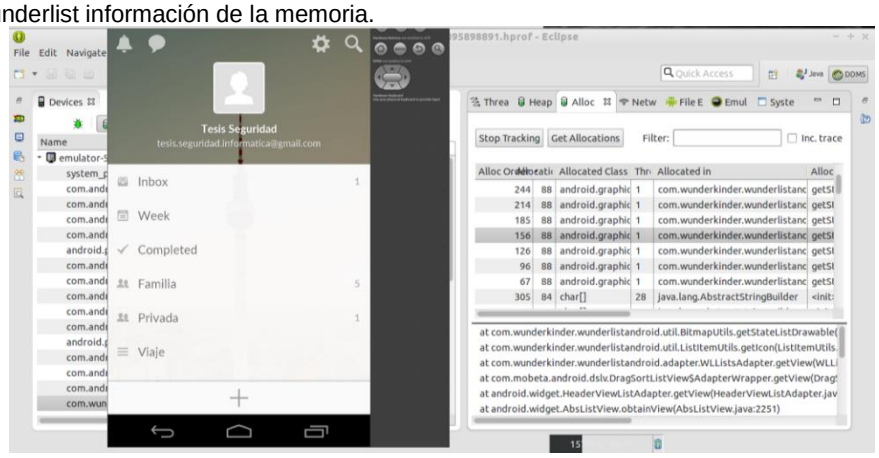


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información sensible de la misma.

Figura 48. Wunderlist información de la memoria.



Fuente: Los Autores.

Figura 49. Wunderlist información de la memoria al cerrar la aplicación.

Fuente: Los Autores.

Figura 50. Wunderlist búsqueda de información en la memoria.

Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación, logrando identificar el id del dispositivo y el id del usuario, lo que representa una vulnerabilidad para la aplicación.

Figura 51. Wunderlist búsqueda de información en el archivo hprof.

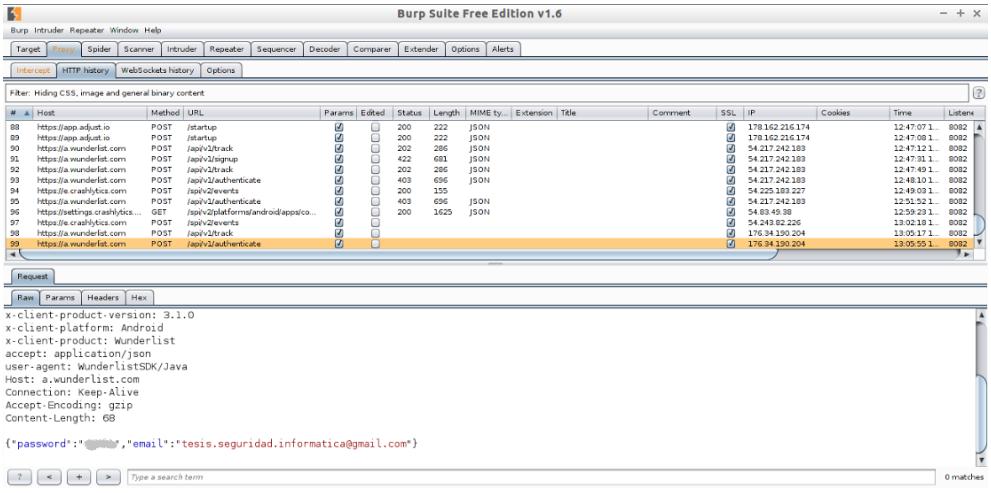
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

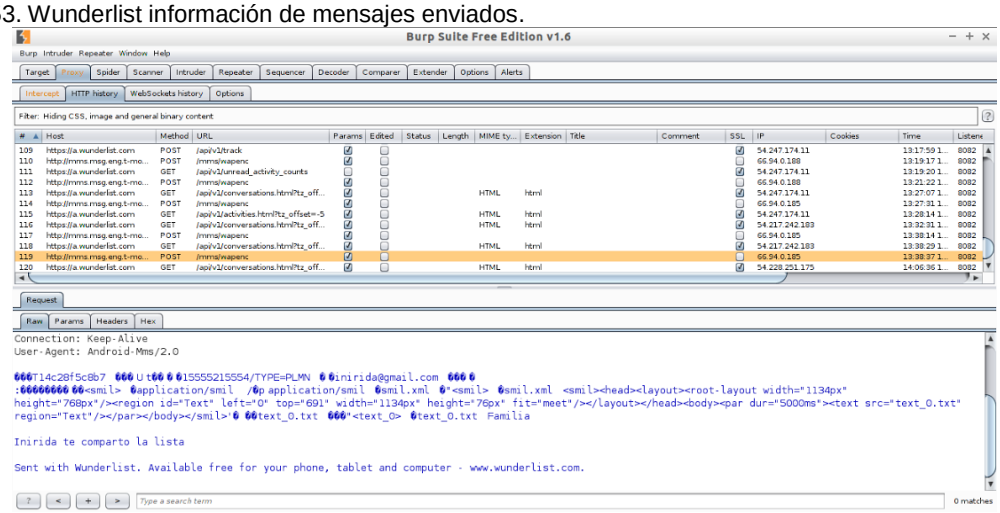
Realizada la interceptación del tráfico de la red se encontró que la aplicación envía información sensible como el usuario y contraseña de acceso a la aplicación, id del usuario, mensajes enviados en claro con la respectiva dirección de correo electrónico sin cifrar, convirtiéndose en una vulnerabilidad en las comunicaciones.

Figura 52. Wunderlist información de autenticación.



Fuente: Los Autores.

Figura 53. Wunderlist información de mensajes enviados.

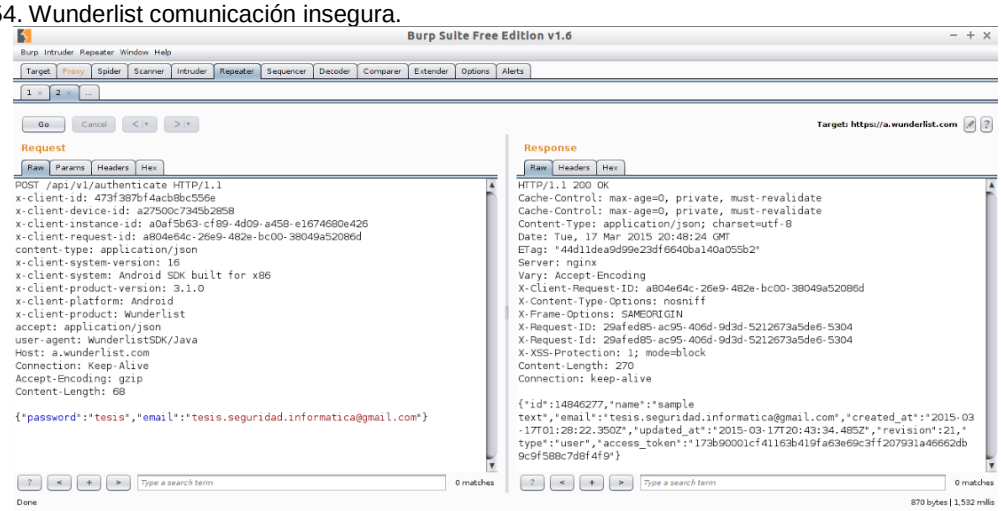


Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

Se observa que los protocolos de comunicación no se usan de forma segura, la aplicación admite comunicación a través de los protocolos HTTP, que envía información sensible como el usuario y contraseña sin cifrar, convirtiéndose en una vulnerabilidad muy alta.

Figura 54. Wunderlist comunicación insegura.



Fuente: Los Autores.

II. Aplicación TED

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
Ted (com.ted.android)

2- Funcionalidad básica
Aplicación oficial de TED que permite presentar debates, charlas o discursos en vídeo o audio (TEDTalk) de expertos sobre diferentes temas, en donde los TEDTalk son actualizados semanalmente. La aplicación se encuentra en traducida en 21 idiomas, permite añadir subtítulos de acuerdo al idioma.

3- ¿La aplicación realiza transacciones electrónicas?

[ ] Si [X] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[ ] Si [X] No

Figura 55. Ted Permisos

Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

[ ] NFC
[ ] GPS
[ ] Micrófono
[X] USB
[ ] Bluetooth
[ ] Cámara
[ ] Sensores

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

[ ] Telefonía (SMS, teléfono) [ ] Contactos

	Recepción de datos de aplicaciones y otros servicios en el dispositivo		Google Wallet
	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)		Correo electrónico
	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 56. Ted interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[ ] Si [X] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[X] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

B- Análisis estático

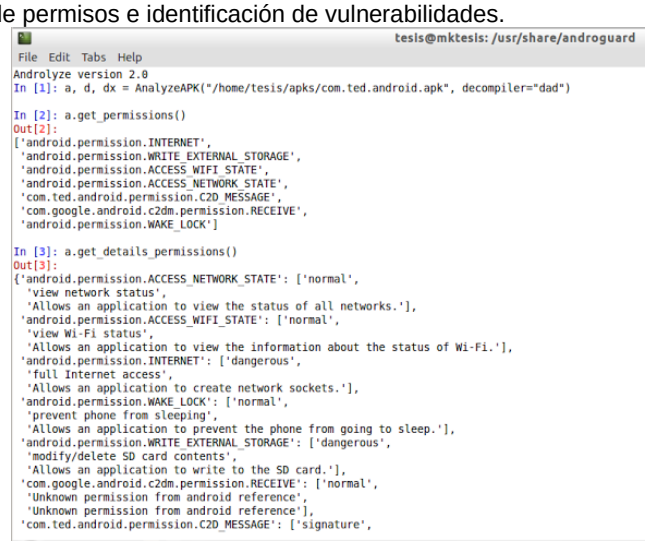
General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.
- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.

Figura 57. Ted revisión de permisos e identificación de vulnerabilidades.



Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

No. Realizada la revisión del código fuente no se encontró uso de métodos de validación de este parámetro en la búsqueda de instrucciones con los comandos “xbin”, “su”, “sbin”, “system”.

Los dispositivos rooteados no incluyen todas las protecciones de seguridad en el sistema operativo permitiendo el acceso total a información y datos de aplicaciones.

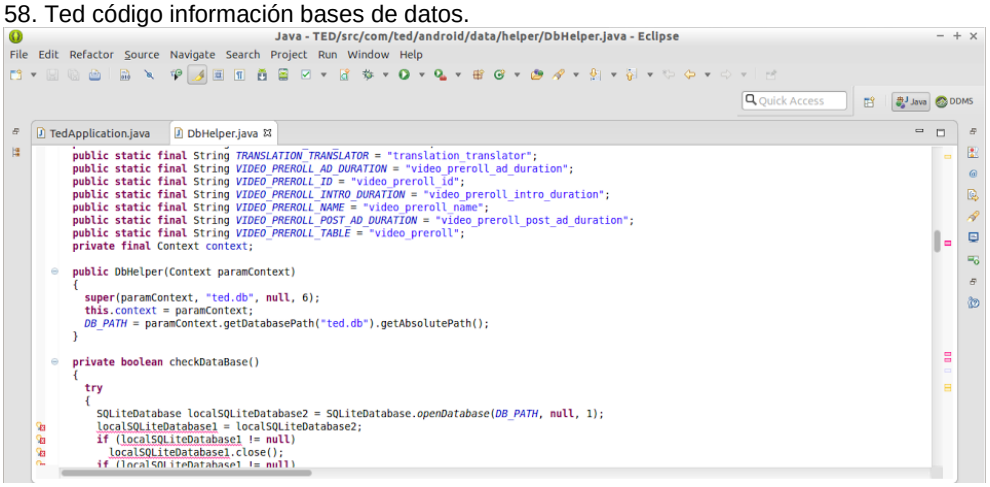
M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa varias bases de datos

- ted.db, información, que se muestra en el archivo *com/ted/android/data/helper/DbHelper.java*, en la cual se almacena la información de las talk.
- ted.db-journal la cual representa datos temporales usados por SQLite.

Figura 58. Ted código información bases de datos.



Fuente: Los Autores.

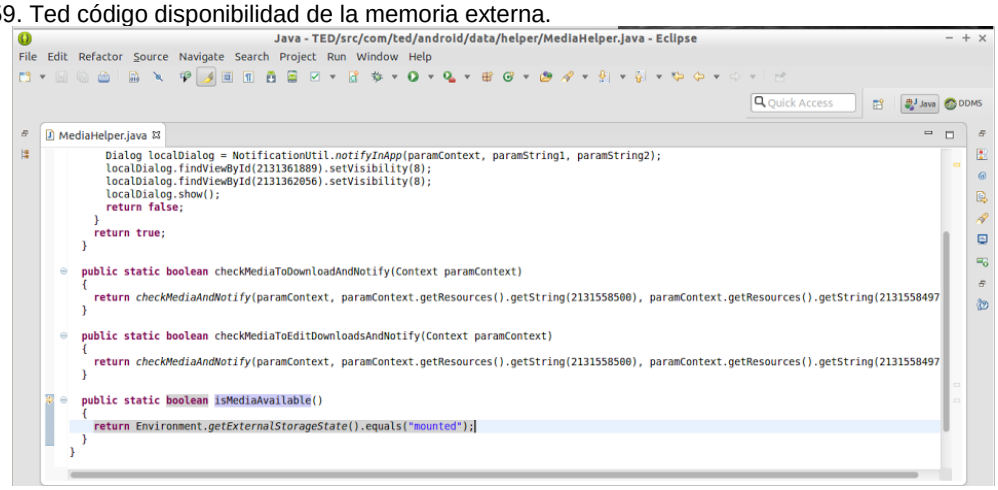
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En la siguiente imagen se muestra el código donde revisa la disponibilidad de la memoria externa.

Figura 59. Ted código disponibilidad de la memoria externa.

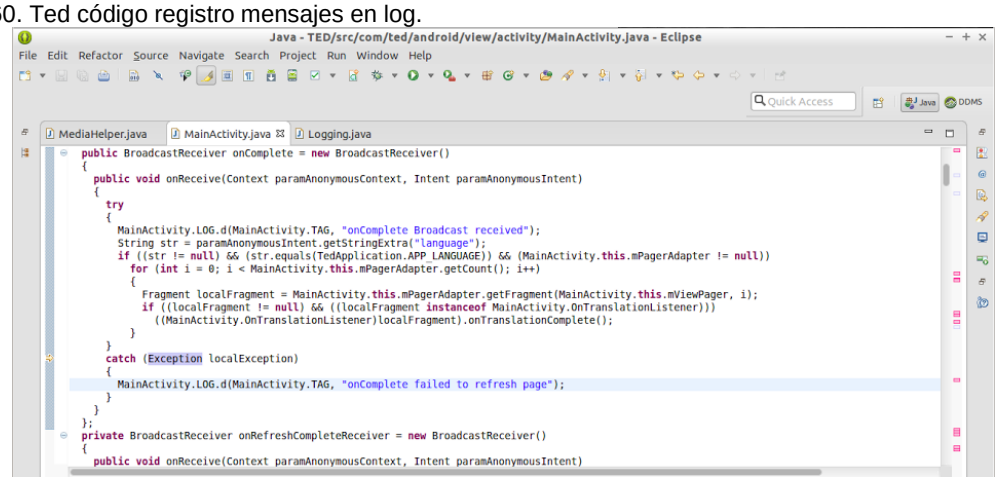


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log pero la información registrada en el log no está cifrada.

Figura 60. Ted código registro mensajes en log.



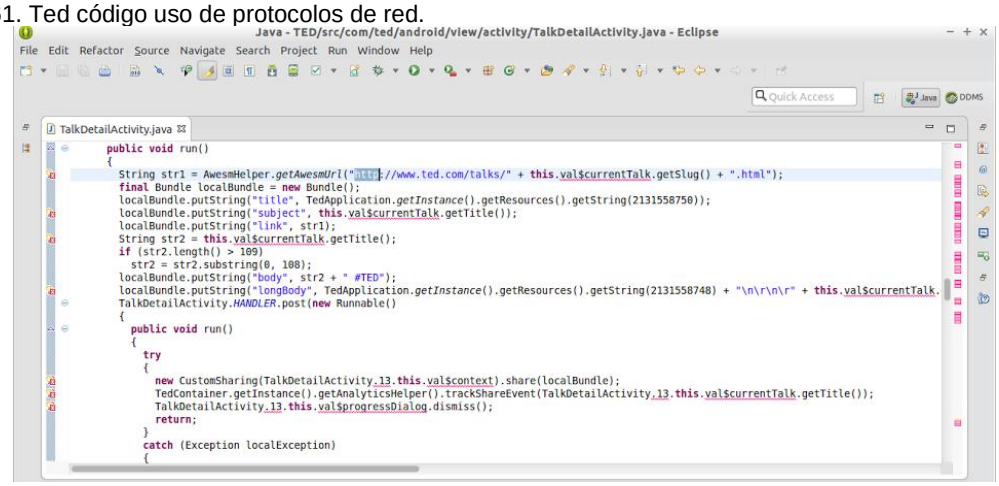
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza solamente protocolo http.

Figura 61. Ted código uso de protocolos de red.

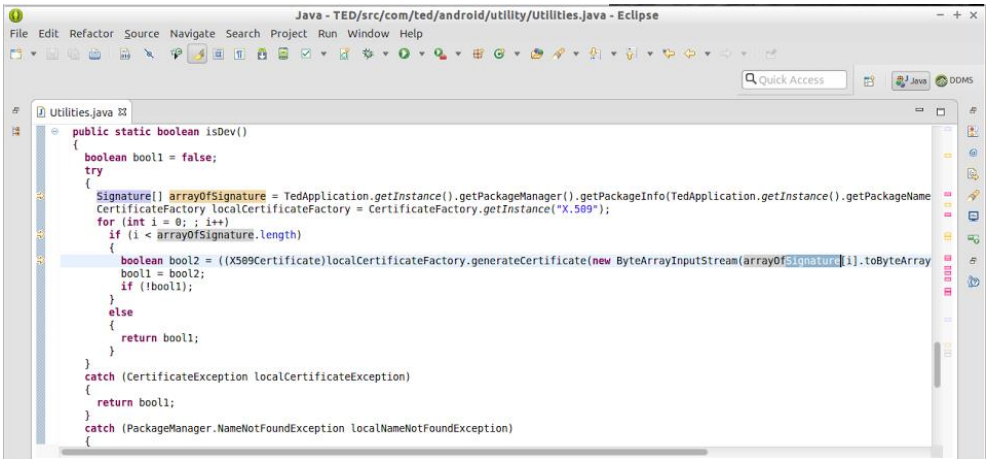


Fuente: Los Autores.

7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

Se realiza verificación de la aplicación en donde se genera el certificado X509.

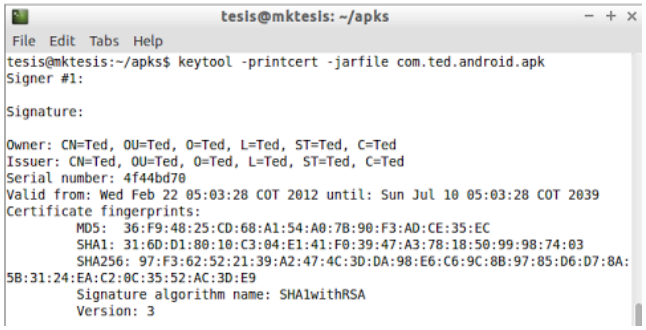
Figura 62. Ted Generación del certificado.



Fuente: Los Autores.

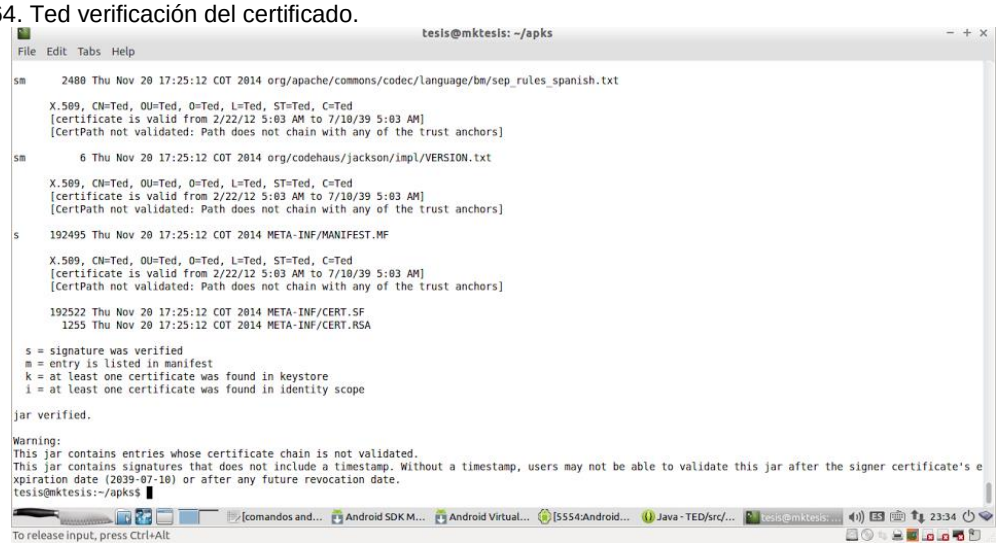
Se encontró que la aplicación utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 63. Ted información del certificado.



Fuente: Los Autores.

Figura 64. Ted verificación del certificado.



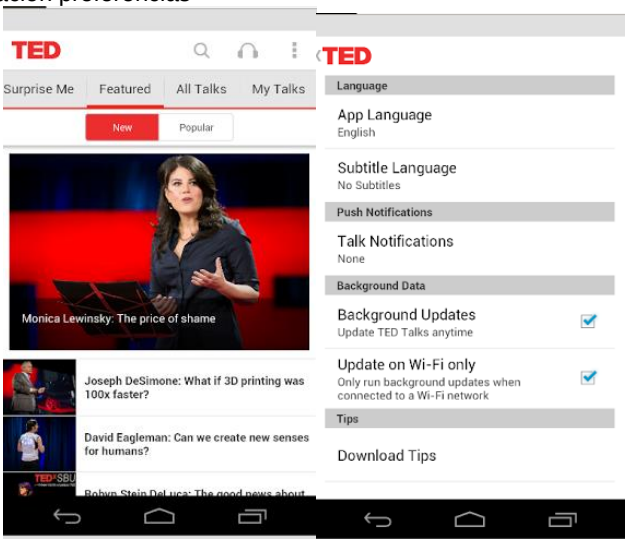
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló y configuró la aplicación, verificando su buen funcionamiento.

Figura 65. Ted configuración preferencias



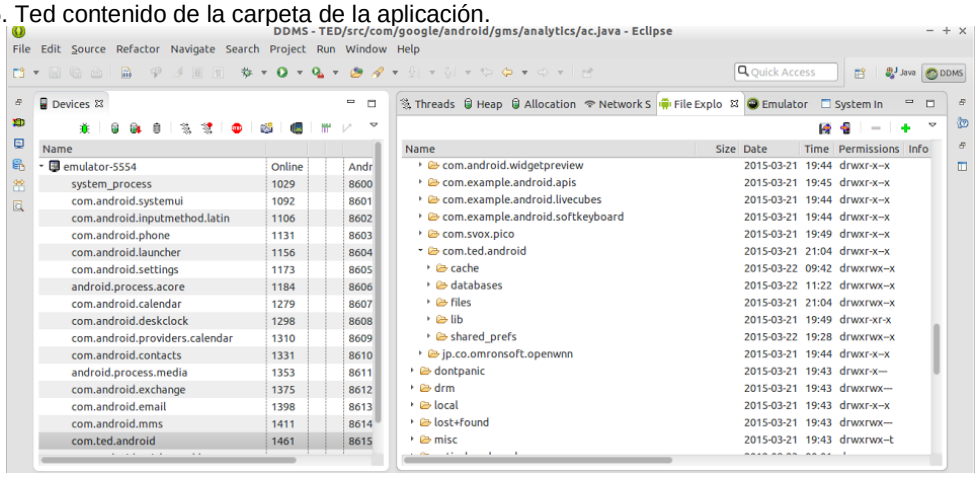
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.ted.android” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

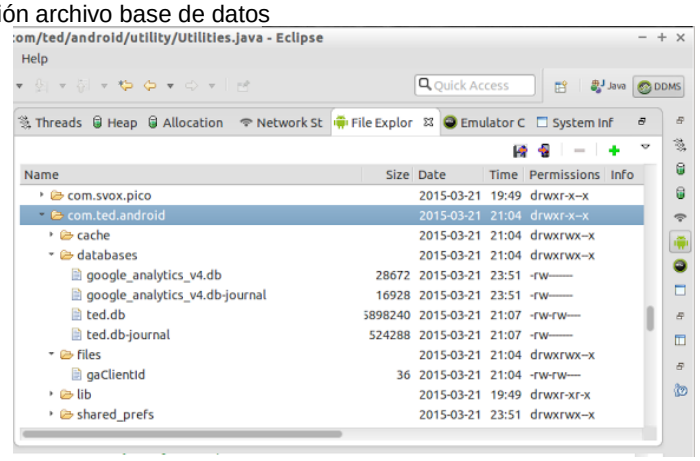
Figura 66. Ted contenido de la carpeta de la aplicación.



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “google_analytics_v4” que pertenece al servicio Google Analytics, “webview.db” para los servicios web y “ted.db” que es la base de datos propia de la aplicación.

Figura 67. Ted ubicación archivo base de datos

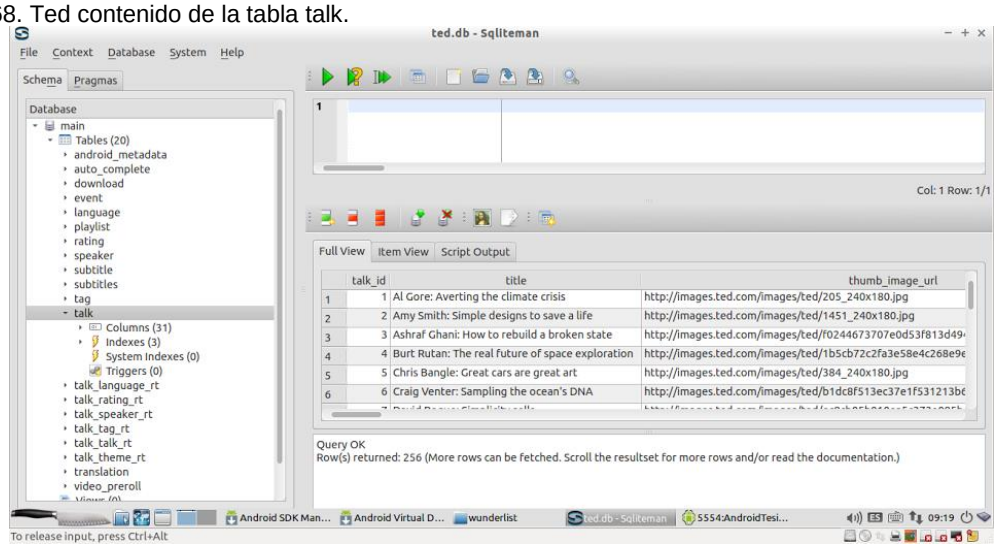


Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “tesd.db” se observan varias tablas entre las cuales se encuentra información para descarga sobre los videos y audios de las charlas que se encuentran en la aplicación, pero no se encuentra información sobre datos sensibles.

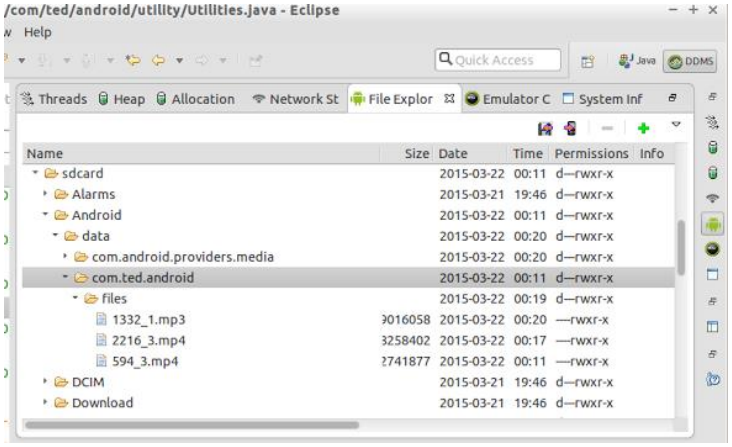
Figura 68. Ted contenido de la tabla talk.



Fuente: Los Autores.

En la carpeta files se encuentran los archivos de audio y video (mp3, mp4) descargados de la aplicación sobre las charlas (talk) que no manejan información sensible. No se observan archivos xml.

Figura 69. Ted Carpeta Files.

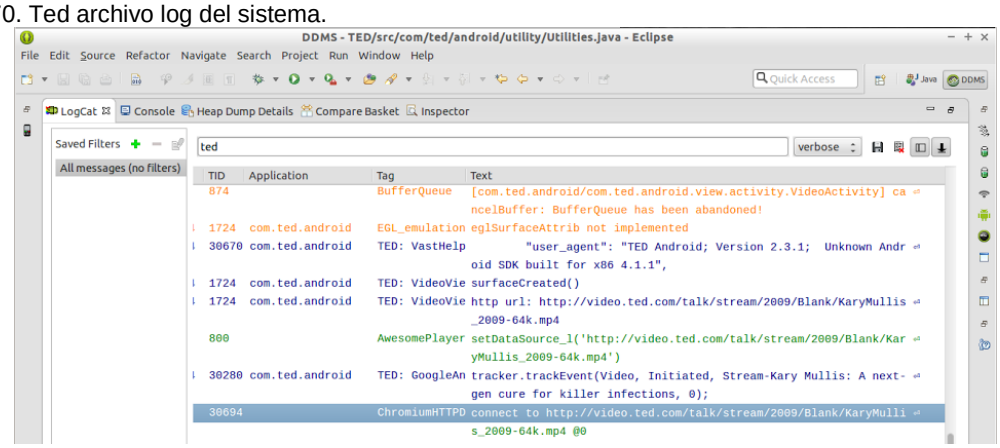


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log no se observa el almacenamiento de información sensible, solo muestra información de las actividades generadas.

Figura 70. Ted archivo log del sistema.

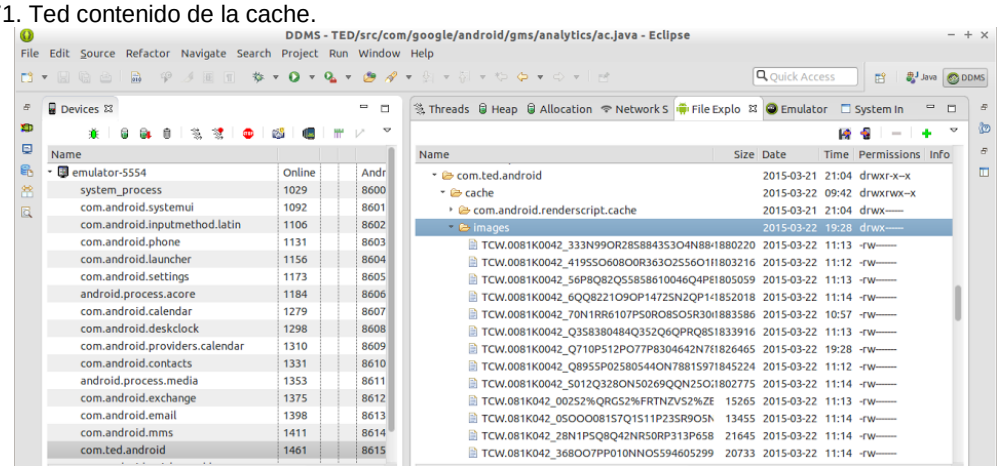


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

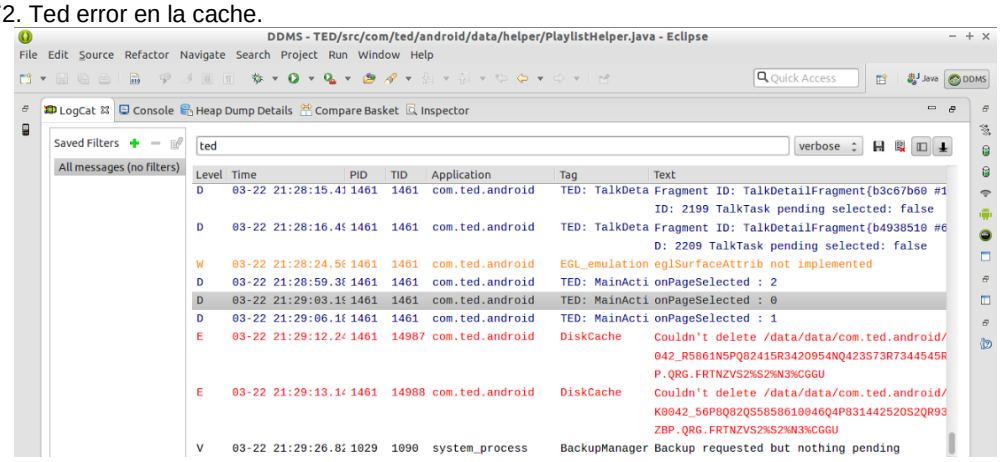
Se procedió a revisar la carpeta cache de la aplicación donde se encontró varias subcarpetas. En la subcarpeta “images” se encuentra el archivo cifrado a través de la aplicación con una función URLEncoder y UTF-8. Estos archivos deben ser borrados por la aplicación pero no se realiza por errores ocurridos en la ejecución como lo muestra el archivo log.

Figura 71. Ted contenido de la cache.



Fuente: Los Autores.

Figura 72. Ted error en la cache.

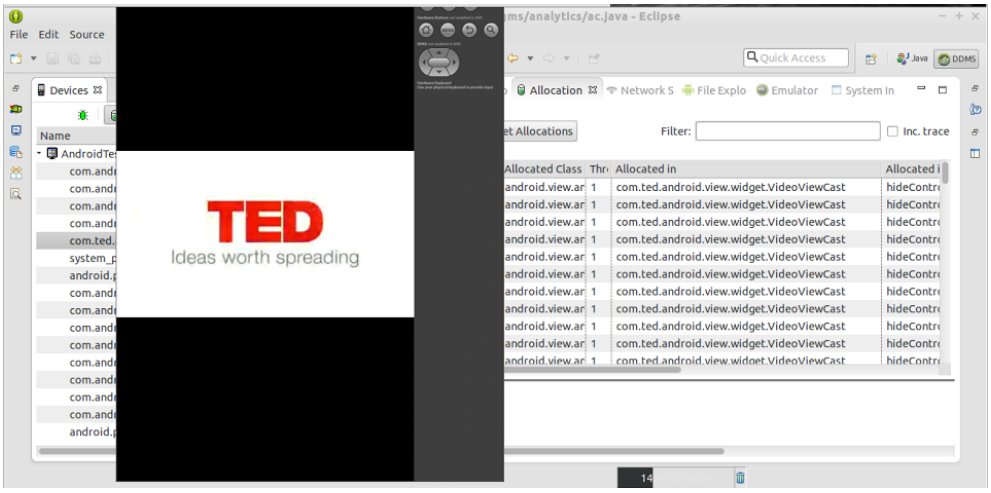


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo y no se identifica el almacenamiento de información sensible abierta o una vez cerrada la aplicación.

Figura 73. Ted información de la memoria.

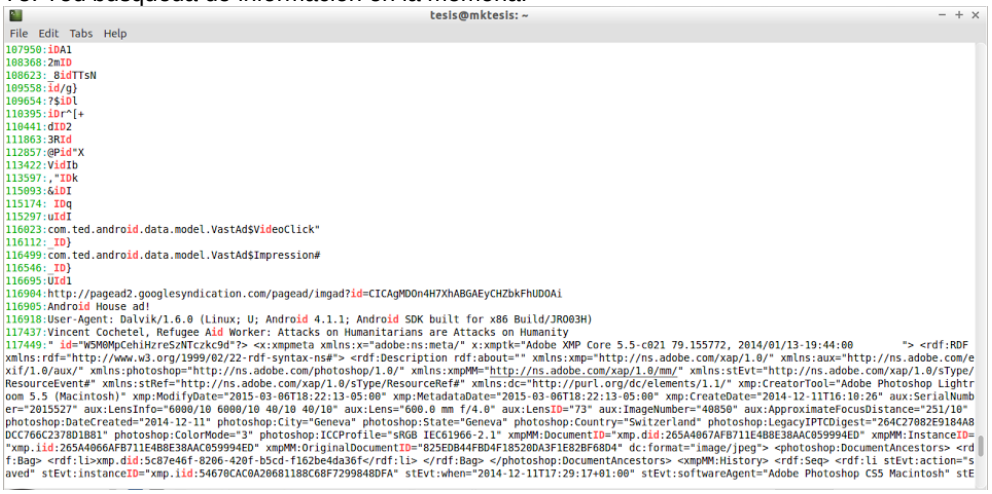


Fuente: Los Autores.

Figura 74. Ted información de la memoria al cerrar la aplicación.

Fuente: Los Autores.

Figura 75. Ted búsqueda de información en la memoria.



Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin encontrarse información sensible.

Figura 76. Ted búsqueda de información en el archivo hprof.

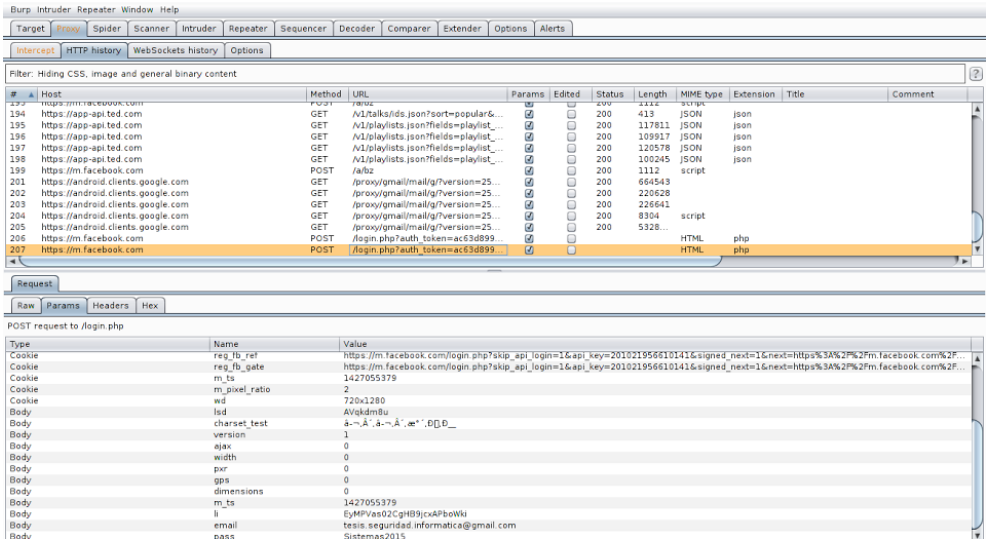
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Realizada la interceptación del tráfico de la red se encontró que la aplicación al compartir información de videos con Facebook usando protocolo HTTPS envía información sensible como el usuario y contraseña de acceso solicitado sin cifrar, convirtiéndose en una vulnerabilidad en las comunicaciones, ya que un atacante puede usar los datos y suplantar la identidad del usuario.

Figura 77. Ted información de autenticación.



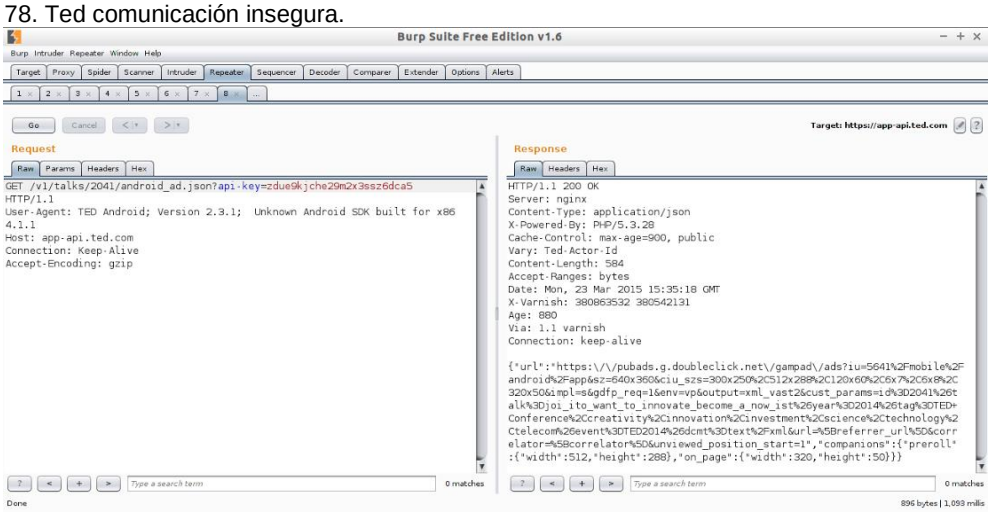
Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

Se observa que los protocolos de comunicación solamente se identifican certificados SSL cuando se conecta a la página http://www.app-api.ted.com/ usando un script json.

En la comunicación a través de los protocolos HTTP, permite el reenvío de petición aun cuando la aplicación ha sido cerrada, pero no se identifica información sensible en la transmisión.

Figura 78. Ted comunicación insegura.



Fuente: Los Autores.

III. Aplicación SwiftKey + Emoji

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
SwiftKey + Emoji (com.touchtype.swiftkey)

2- Funcionalidad básica

Aplicación que ofrece la mejor predicción de la siguiente palabra, autocorrección inteligente, compatibilidad con más de 800 emoticonos, predicción de emoticonos, escritura rápida de SMS, chat, texto y correo electrónico.

3- ¿La aplicación realiza transacciones electrónicas?

[X] Si [ ] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[X] Si [ ] No

Figura 79. SwiftKey permisos

Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

[ ]	NFC	[ ]	Bluetooth
[ ]	GPS	[ ]	Cámara
[ ]	Micrófono	[ ]	Sensores
[X]	USB		

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

[X]	Telefonía (SMS, teléfono)	[ ]	Contactos
[ ]	Recepción de datos de aplicaciones y otros servicios en el dispositivo	[ ]	Google Wallet
[ ]	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)	[ ]	Correo electrónico
[ ]	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 80. SwiftKey interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[ ] Si [X] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[X] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

B- Análisis estático

General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

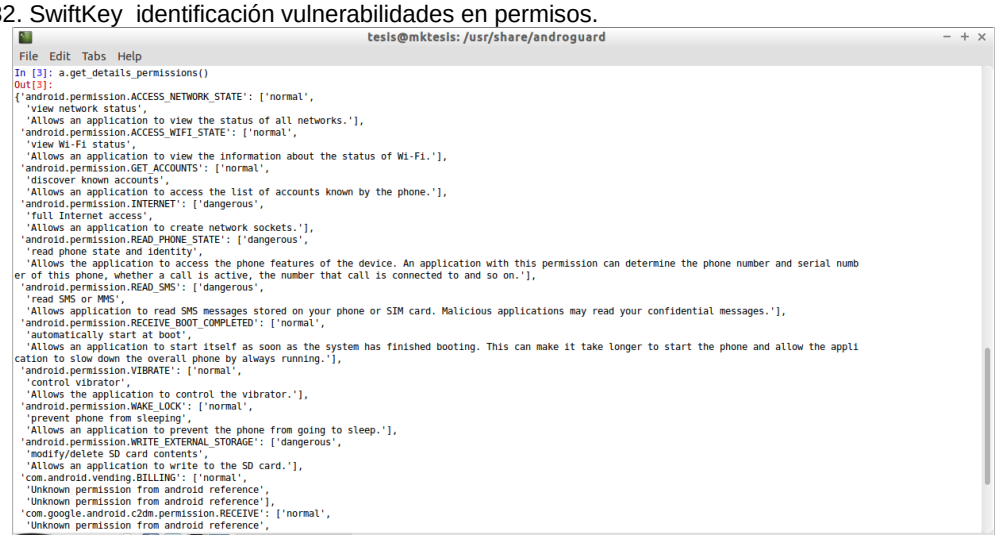
El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.
- READ_PHONE_STATE: permite acceder a las funciones del teléfono del dispositivo. Una aplicación con estos permisos puede determinar el número y serial del dispositivo, cuando una llamada esta activa puede obtener la información del número al que llama y así sucesivamente.
- READ_SMS: permite leer mensajes guardados en la simcard del teléfono. Las aplicaciones maliciosas pueden leer los mensajes confidenciales.
- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.
- READ_CONTACTS permite el acceso a la información de contactos del dispositivo, sin poder controlar que destino se le dará a estos datos.

Figura 81. SwiftKey revisión de permisos.

Fuente: Los Autores.

Figura 82. SwiftKey identificación vulnerabilidades en permisos.

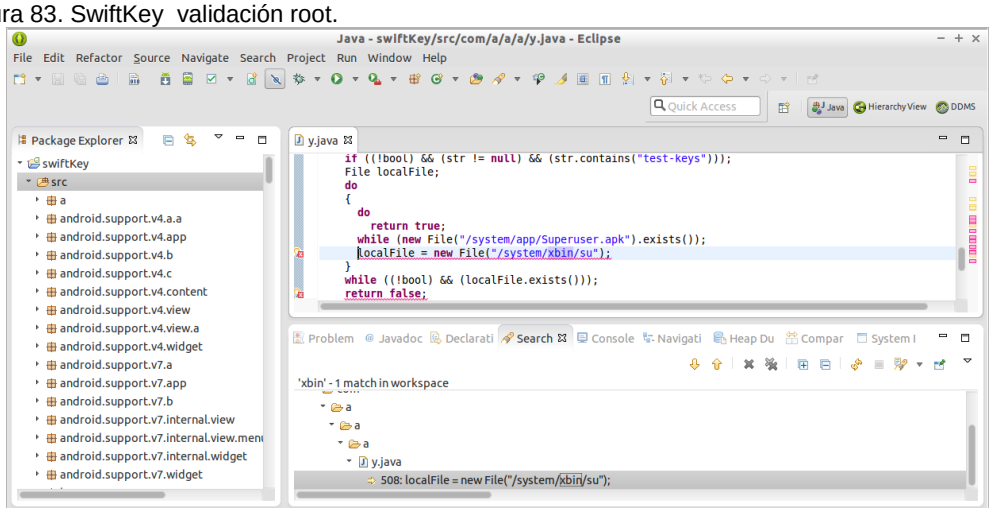


Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

Si, valida la existencia de la aplicación “Superuser.apk” para determinar si el dispositivo esta rooteado.

Figura 83. SwiftKey validación root.



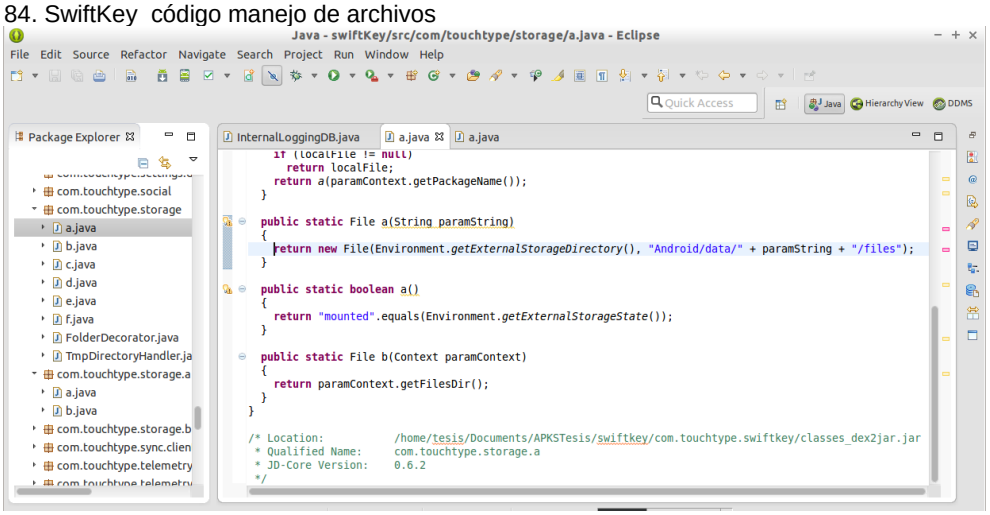
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación no maneja una base de datos de contenido, crea archivos para almacenar información en la ejecución en sdcard en la carpeta /Android/Data, el archivo que muestra esta información es: *com/touchtype/storage/a.java*

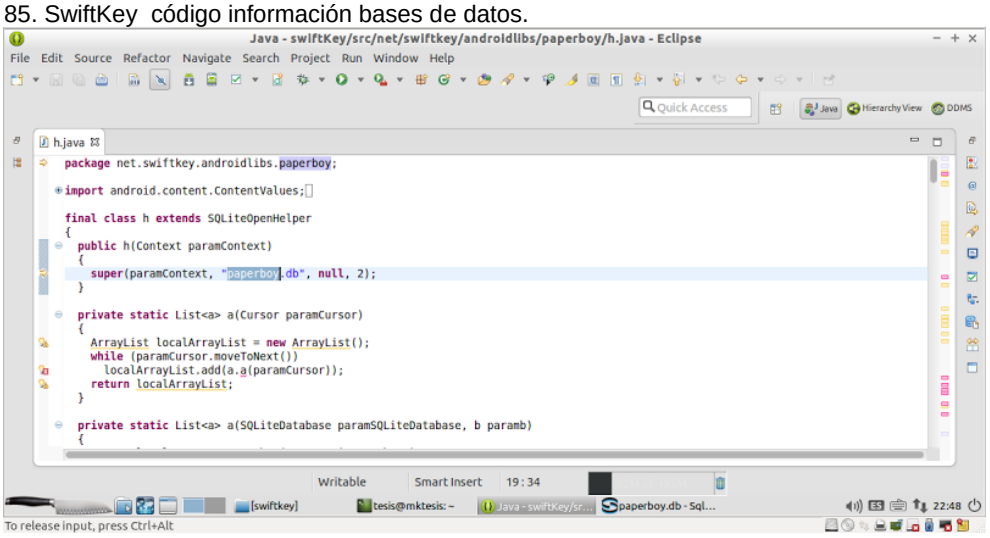
Figura 84. SwiftKey código manejo de archivos



Fuente: Los Autores.

Sin embargo, se identifica la base de datos “paperboy.db” que se encuentra en el archivo *net/swiftkey/androidlibs/paperboy/h.java* la cual maneja una estructura simple para metadata.

Figura 85. SwiftKey código información bases de datos.



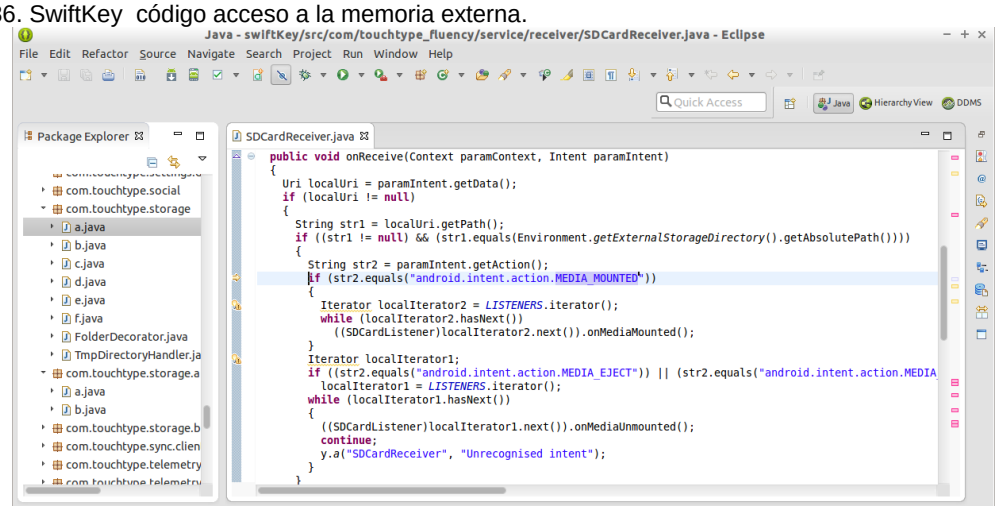
Fuente: Los Autores.

4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones. En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 86. SwiftKey código acceso a la memoria externa.

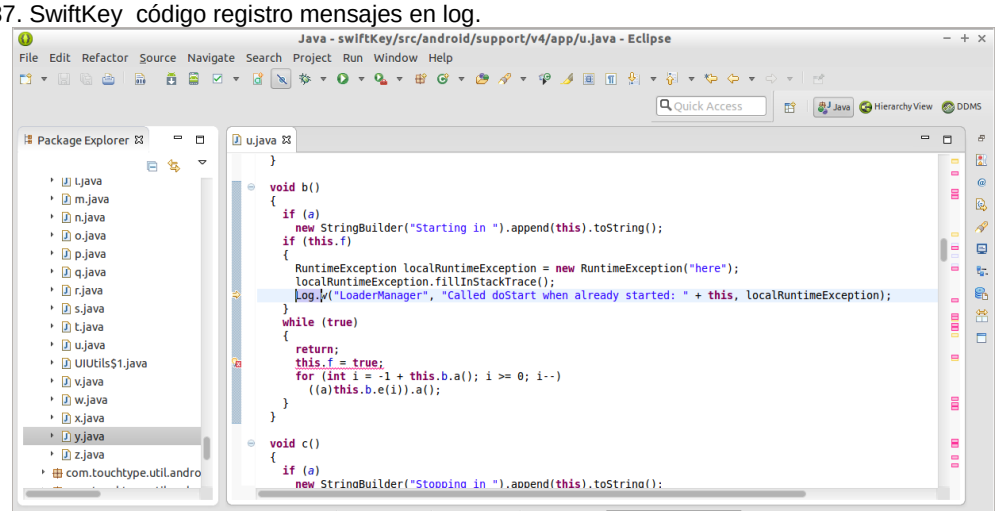


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 87. SwiftKey código registro mensajes en log.



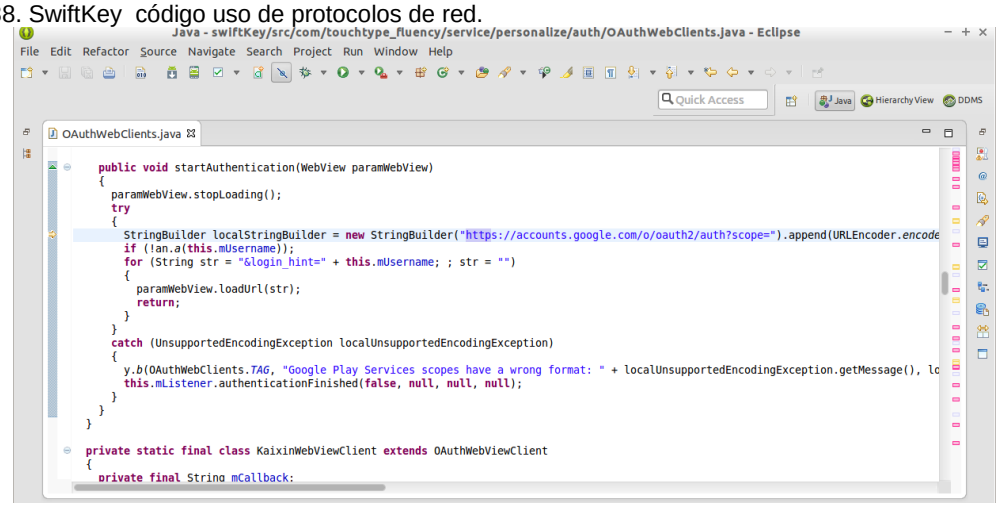
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 88. SwiftKey código uso de protocolos de red.



Fuente: Los Autores.

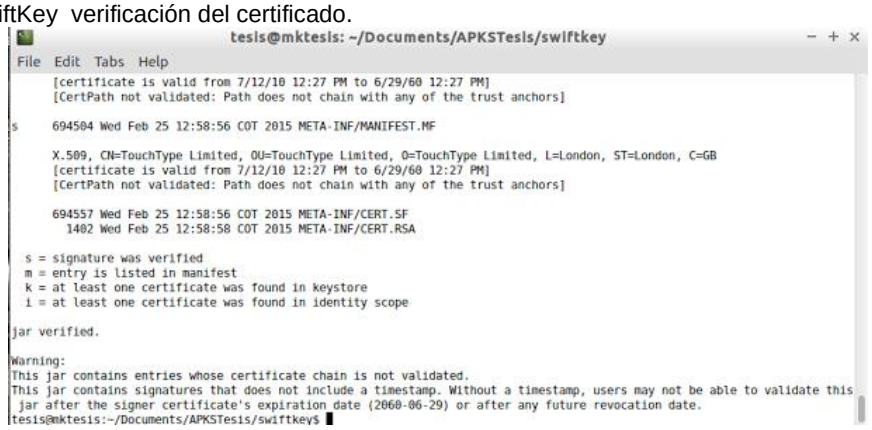
7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 89. SwiftKey información del certificado.

Fuente: Los Autores.

Figura 90. SwiftKey verificación del certificado.



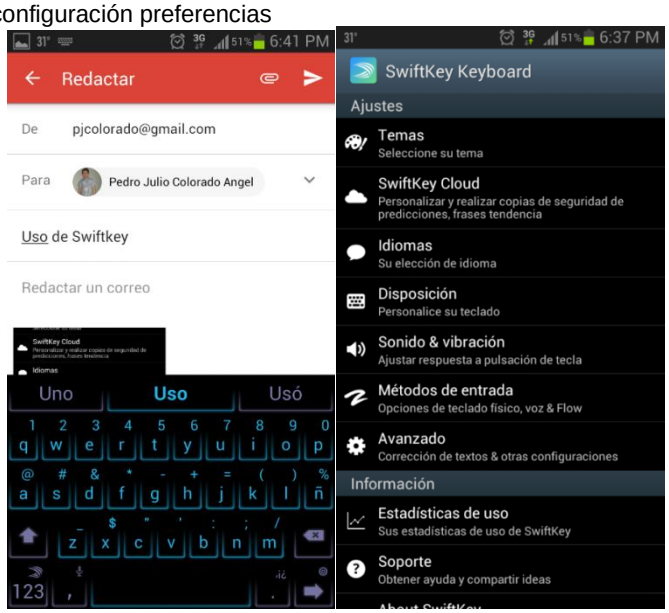
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 91. SwiftKey configuración preferencias



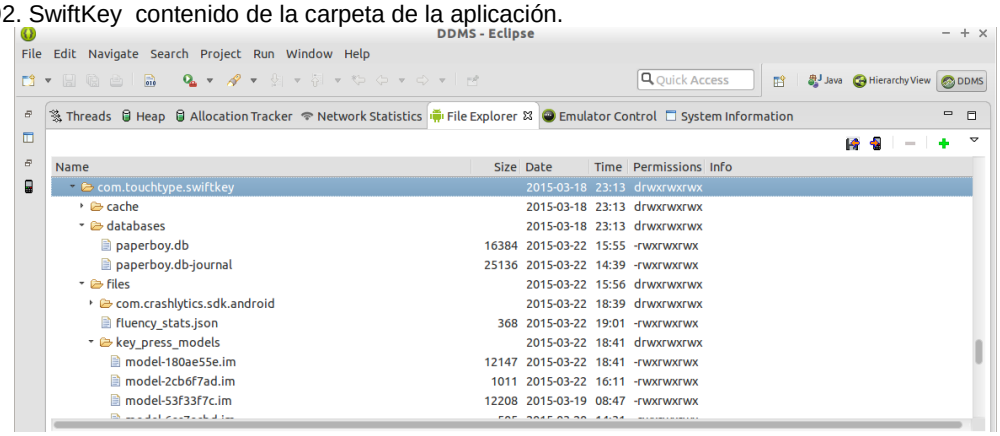
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.touchtype.swiftkey” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

Figura 92. SwiftKey contenido de la carpeta de la aplicación.



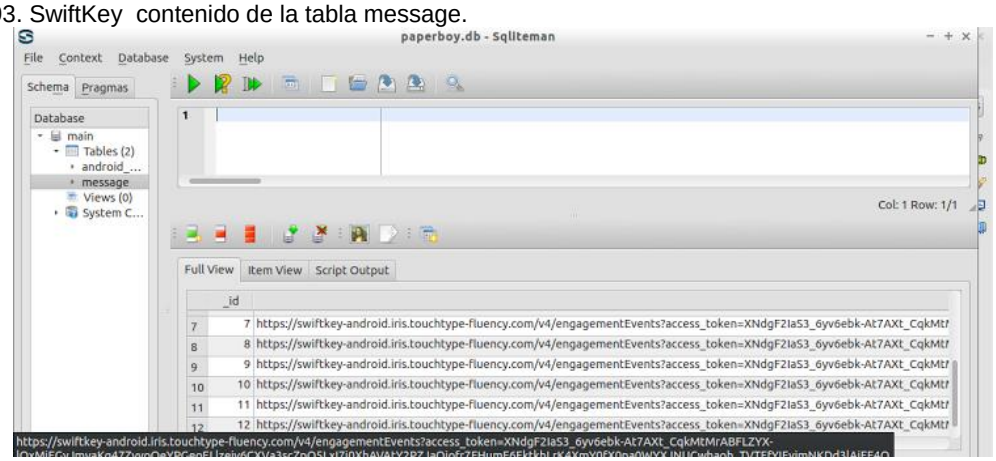
Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de la base de datos “paperboy.db” y en la carpeta “files” sub carpeta “key_press_models” se manejan los archivos de la aplicación.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “paperboy.db” no contiene información sensible, se observan dos tablas “android_metadata” que contiene la localización y “message” contiene mensajes de conexión a *url* con los datos codificados.

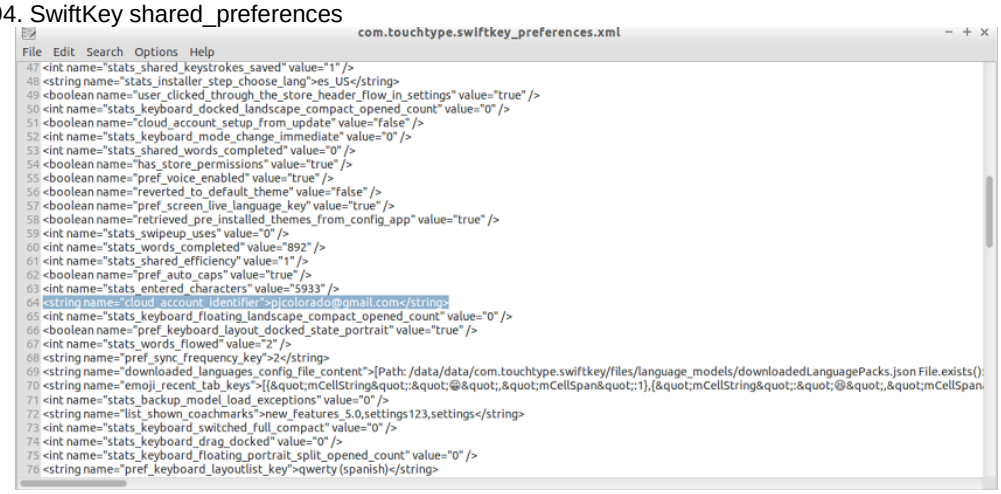
Figura 93. SwiftKey contenido de la tabla message.



Fuente: Los Autores.

En la carpeta shared_preferences se encuentran particularmente el archivo “com.touchtype.swiftkey_preferences.xml” el cual contiene la configuración de los parámetros de la aplicación y se identifica el correo electrónico ingresado para enlace a la nube.

Figura 94. SwiftKey shared_preferences

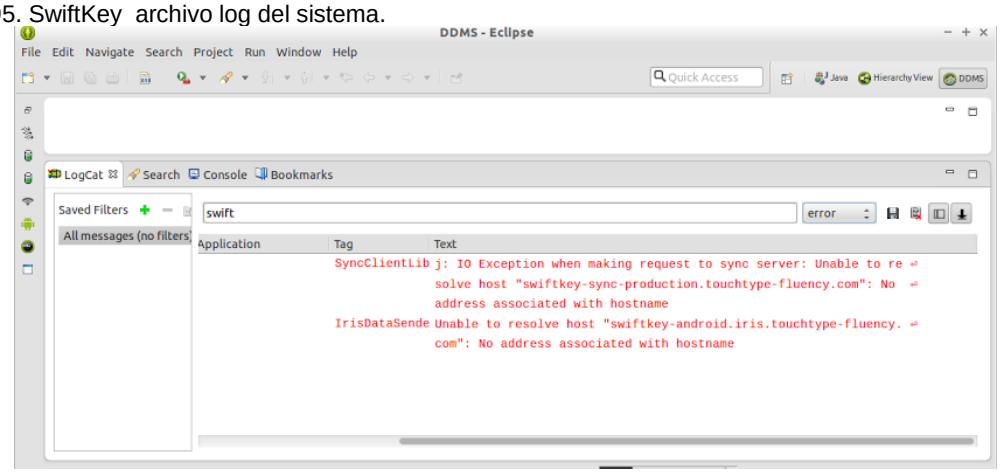


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log no se observa el almacenamiento de información sensible, solo muestra información de las actividades generadas.

Figura 95. SwiftKey archivo log del sistema.

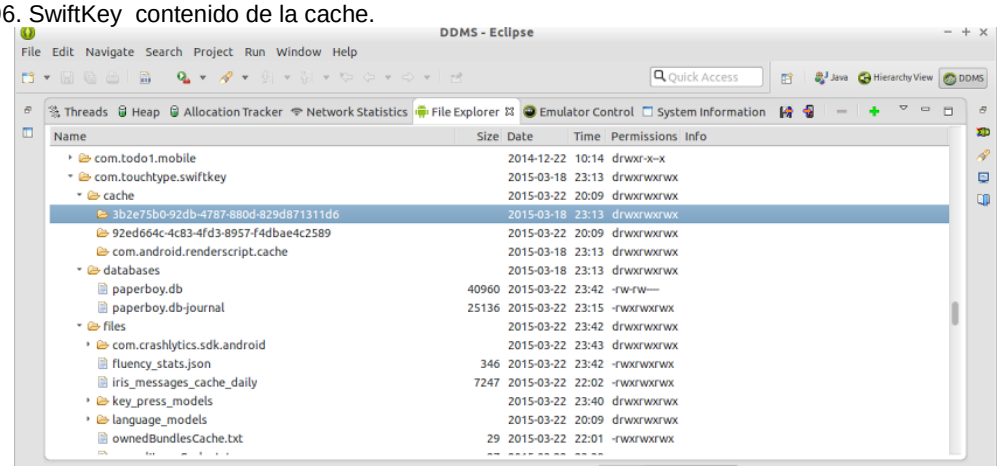


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación en donde se encontraron varias subcarpetas sin contenido.

Figura 96. SwiftKey contenido de la cache.

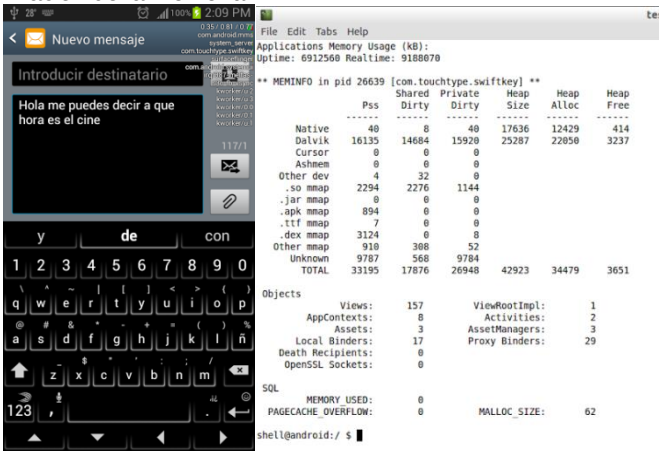


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información de la misma.

Figura 97. SwiftKey información de la memoria.



Fuente: Los Autores.

Figura 98. SwiftKey actividades que se ejecutan en memoria.

Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

No es posible realizar un volcado de la memoria, por cuanto el proceso se encuentra en foreground.

Figura 99. SwiftKey búsqueda de información en la memoria.

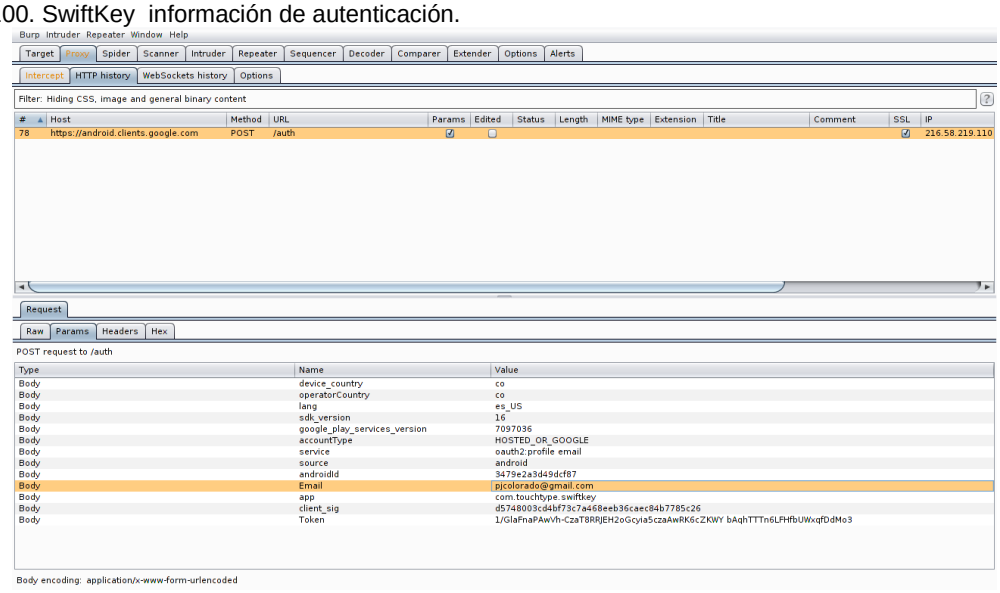
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Realizada la interceptación del tráfico HTTPS de la red se encontró que la aplicación envía información sensible como la dirección de correo electrónico del usuario al loggearse a la cuenta de cloud de swiftkey, convirtiéndose en una vulnerabilidad en las comunicaciones.

Figura 100. SwiftKey información de autenticación.

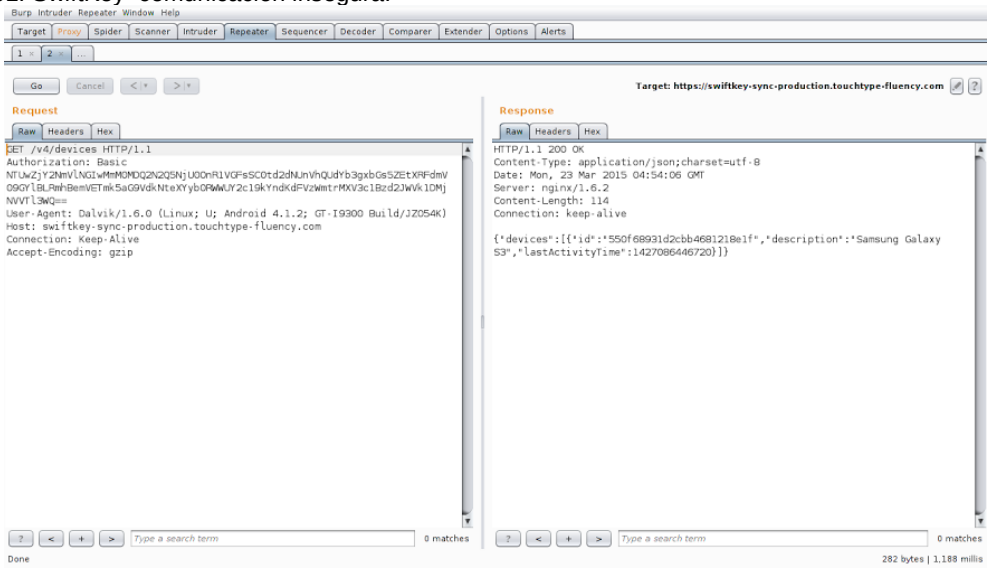


Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

Se observa que los protocolos de comunicación no se usan de forma segura que no se identifican certificados SSL y se permite el reenvío de peticiones aun cuando la cesión actual ha sido terminada, aun cuando no se identifica el envío de envío de información sensible.

Figura 101. SwiftKey comunicación insegura.



Fuente: Los Autores.

IV. Aplicación Lumosity

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
Lumosity (com.lumoslabs.lumosity)

2- Funcionalidad básica
Es una herramienta educativa diseñada por científicos para entrenamiento cerebral, desarrollando habilidades cognitivas, ejercitando la memoria y atención.

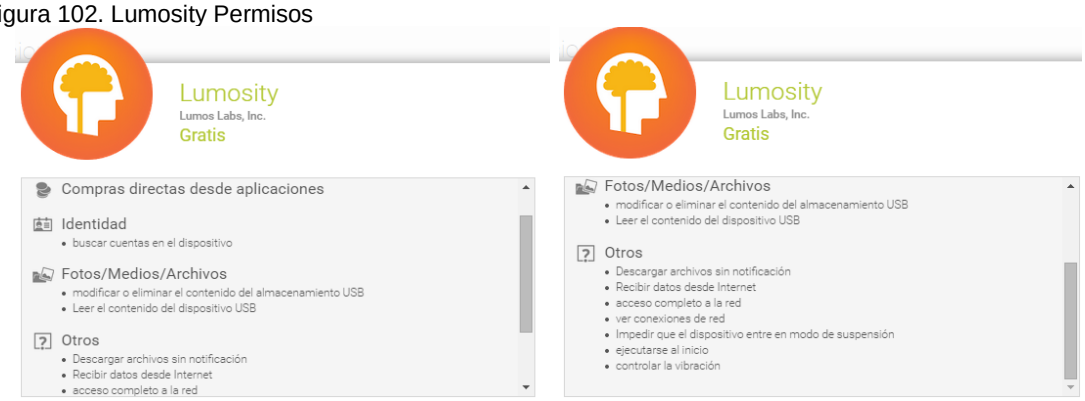
3- ¿La aplicación realiza transacciones electrónicas?

[X] Si [ ] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[x] Si [ ] No

Figura 102. Lumosity Permisos



Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

[ ] NFC [ ] Bluetooth
[ ] GPS [ ] Cámara
[ ] Micrófono [ ] Sensores
[X] USB

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

	Telefonía (SMS, teléfono)		Contactos
	Recepción de datos de aplicaciones y otros servicios en el dispositivo		Google Wallet
X	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)		Correo electrónico
	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 103. Lumosity interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[X] Si [ ] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[ ] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

B- Análisis estático

General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

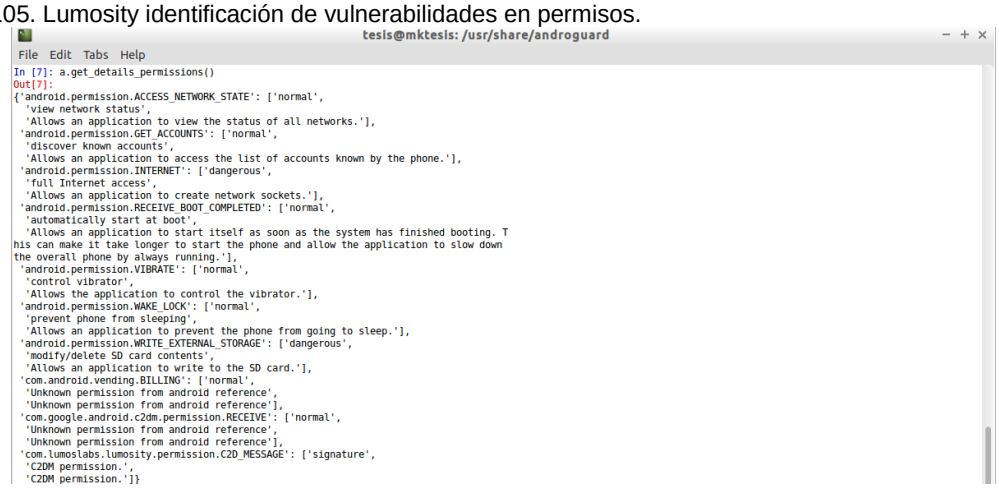
El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.
- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.

Figura 104. Lumosity revisión de permisos.

Fuente: Los Autores.

Figura 105. Lumosity identificación de vulnerabilidades en permisos.



Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

No. Realizada la revisión del código fuente no se encontró uso de métodos de validación de este parámetro en la búsqueda de instrucciones con los comandos “xbin”, “su”, “sbin”, “system”.

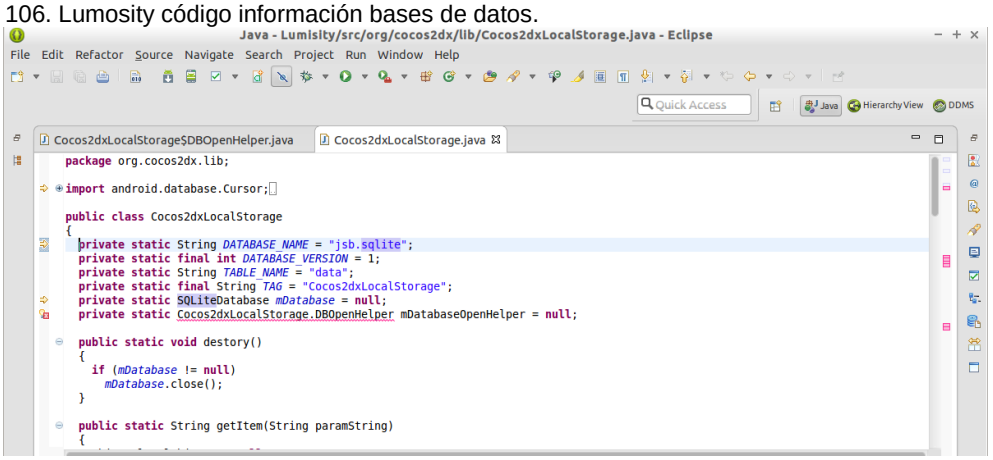
Los dispositivos rooteados no incluyen todas las protecciones de seguridad en el sistema operativo permitiendo el acceso total a información y datos de aplicaciones.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación utiliza una base de datos jsb.sqlite, información, que se muestra en el archivo *org/cocos2dx/lib/Cocos2dxLocalStorage.java*.

Figura 106. Lumosity código información bases de datos.



Fuente: Los Autores.

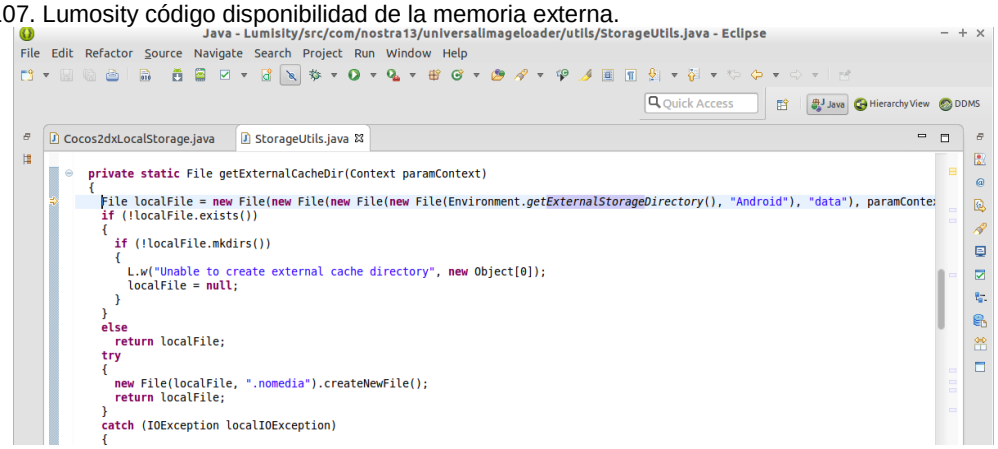
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En la siguiente imagen se muestra el código donde se accede a la memoria externa.

Figura 107. Lumosity código disponibilidad de la memoria externa.

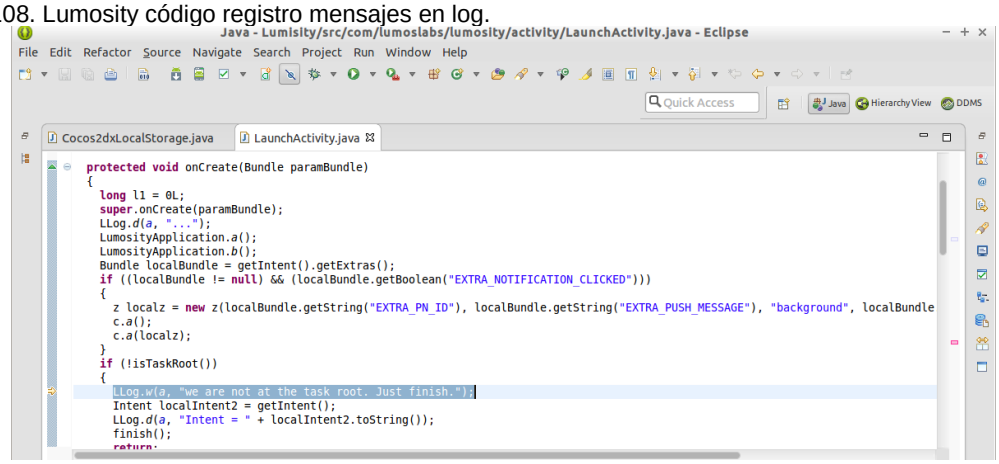


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log pero la información registrada en el log no está cifrada.

Figura 108. Lumosity código registro mensajes en log.



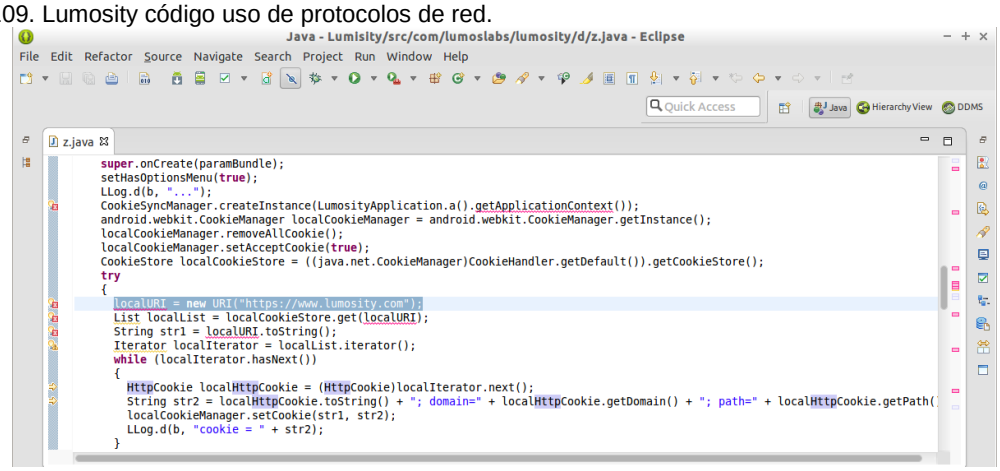
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza solamente protocolo http.

Figura 109. Lumosity código uso de protocolos de red.



Fuente: Los Autores.

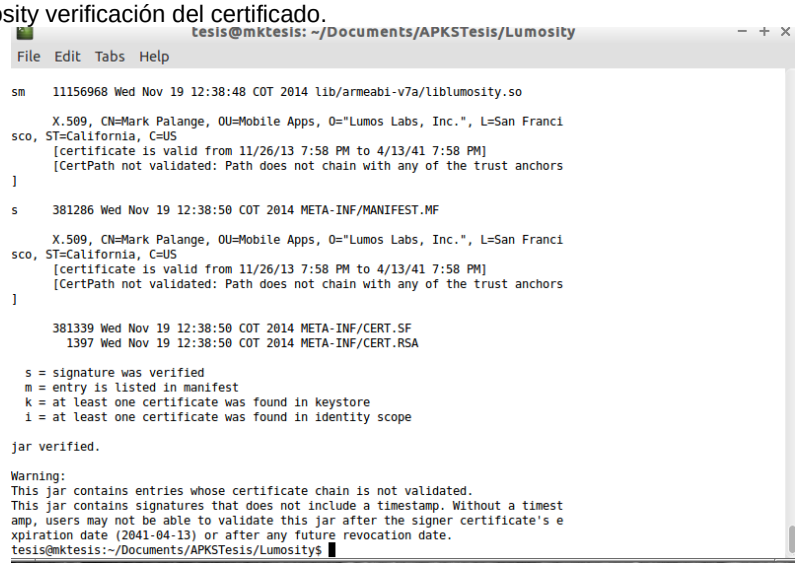
7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

Se encontró que la aplicación utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 110. Lumosity información del certificado.

Fuente: Los Autores.

Figura 111. Lumosity verificación del certificado.



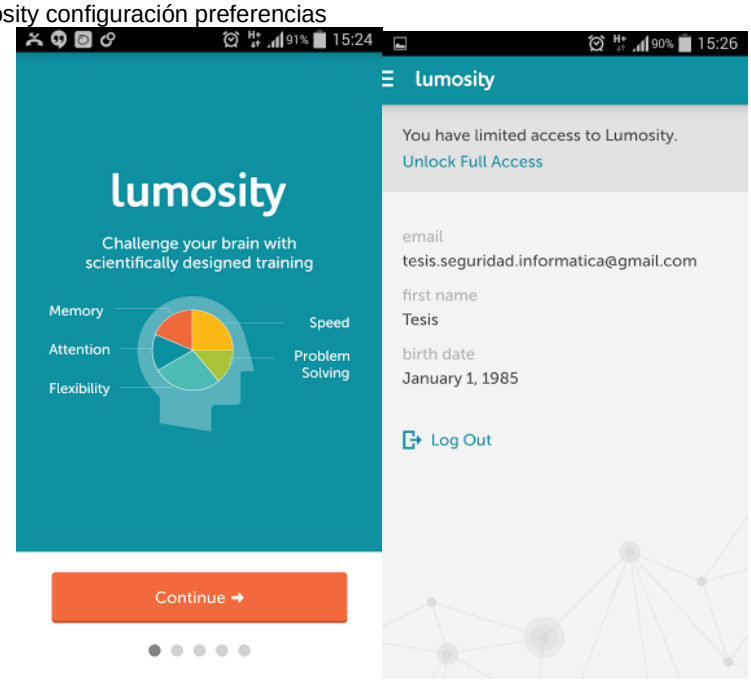
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló y configuró la aplicación, verificando su buen funcionamiento.

Figura 112. Lumosity configuración preferencias



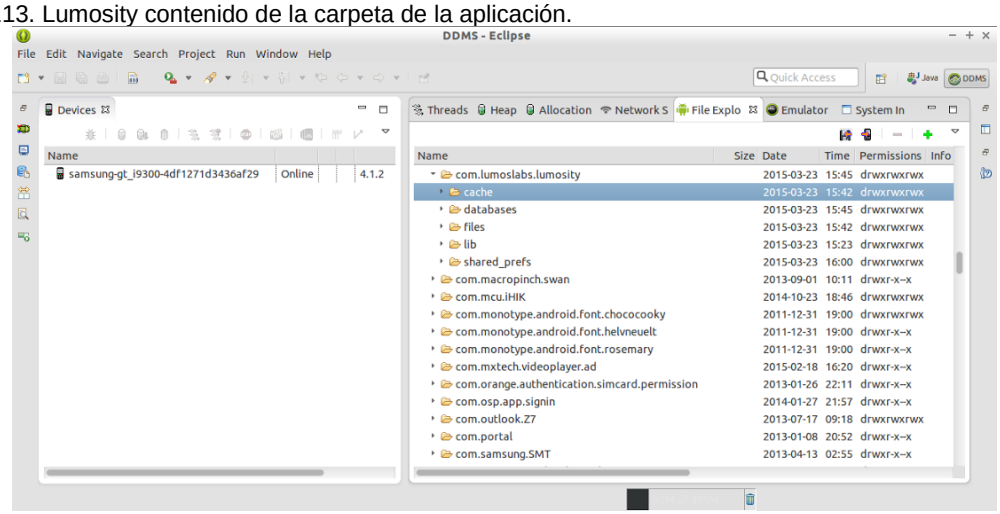
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.Lumosity.android” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

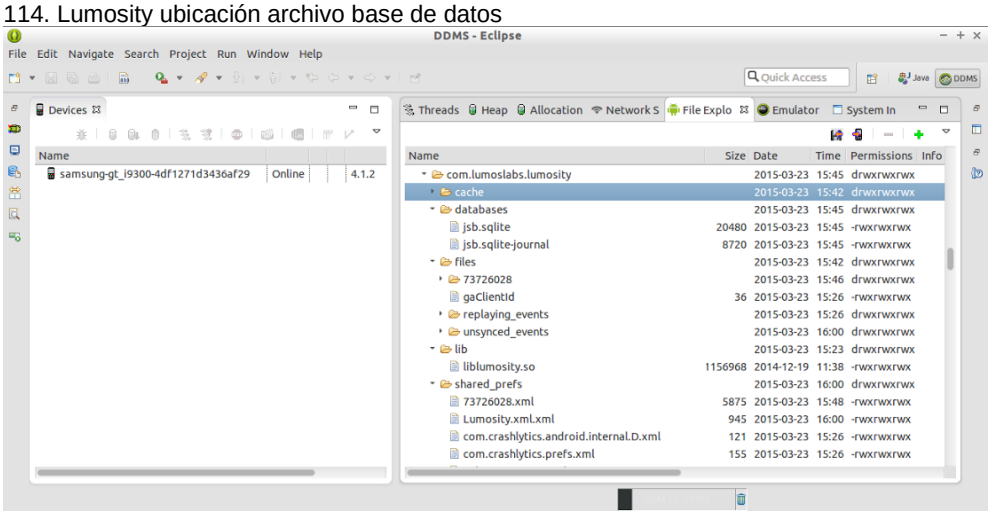
Figura 113. Lumosity contenido de la carpeta de la aplicación.



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de la base de dato “jsb.sqlite”.

Figura 114. Lumosity ubicación archivo base de datos

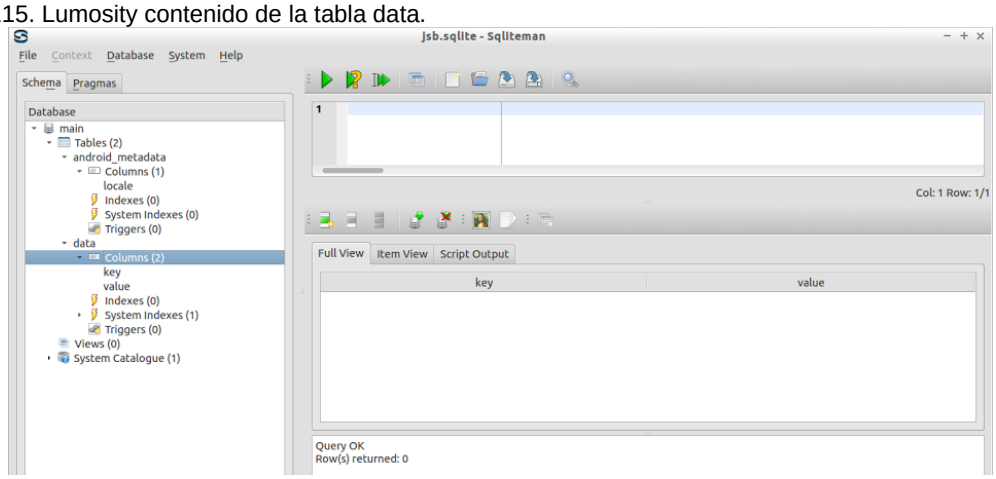


Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “jsb.sqlite” se observa dos tablas, pero no se encuentra información sobre datos sensibles.

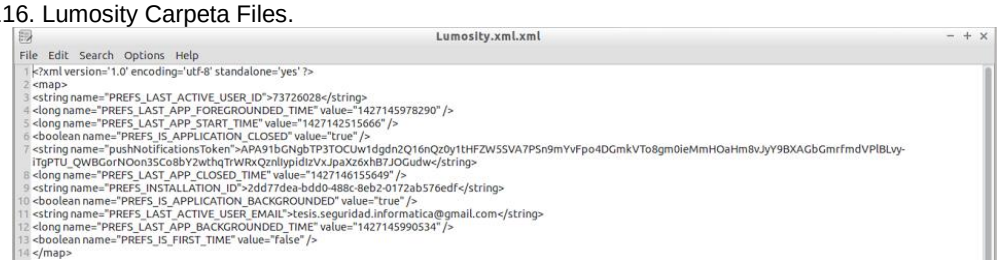
Figura 115. Lumosity contenido de la tabla data.



Fuente: Los Autores.

En la carpeta files se encuentran archivos xml que contienen información sensible como es el identificador del usuario y el correo electrónico.

Figura 116. Lumosity Carpeta Files.

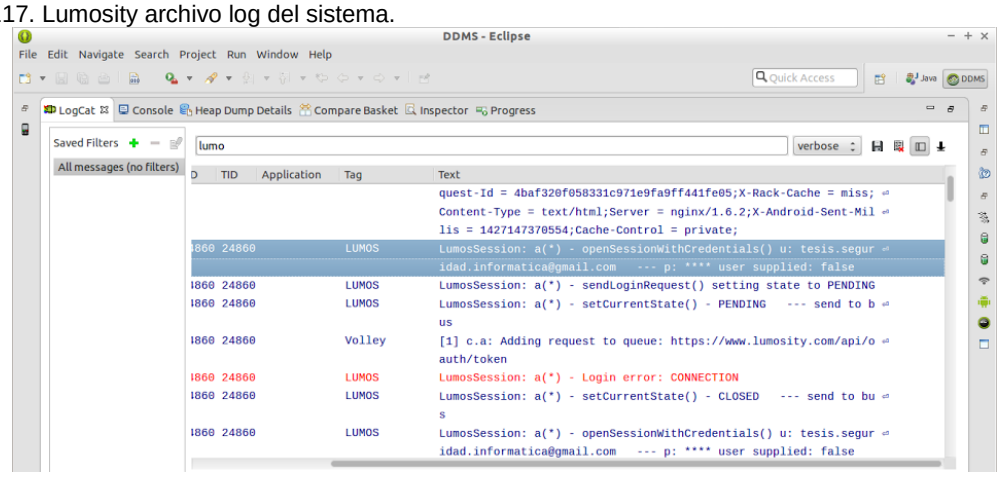


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log se observa el almacenamiento de información sensible como es el usuario del correo electrónico para loggearse en la aplicación.

Figura 117. Lumosity archivo log del sistema.

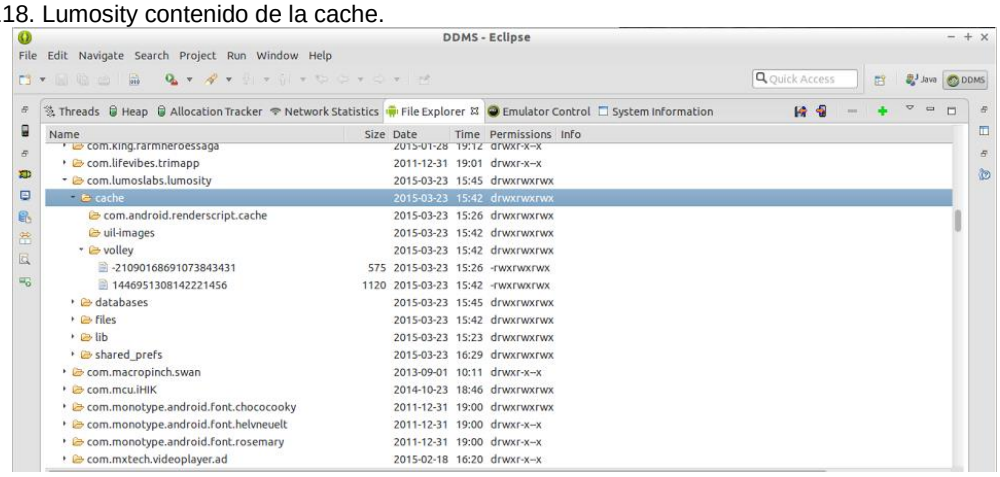


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación sin encontrar información sensible.

Figura 118. Lumosity contenido de la cache.

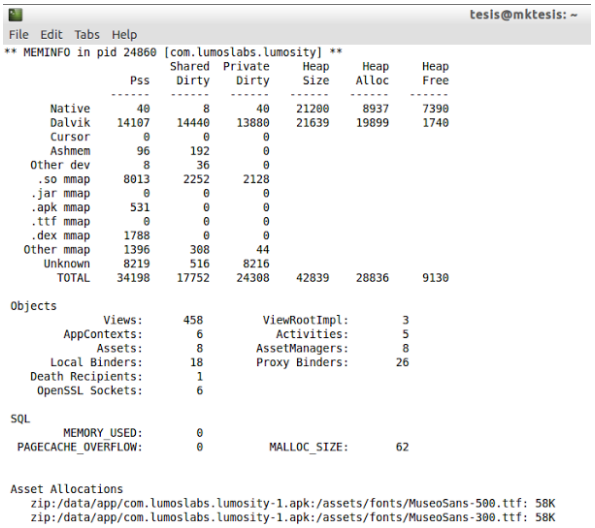


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

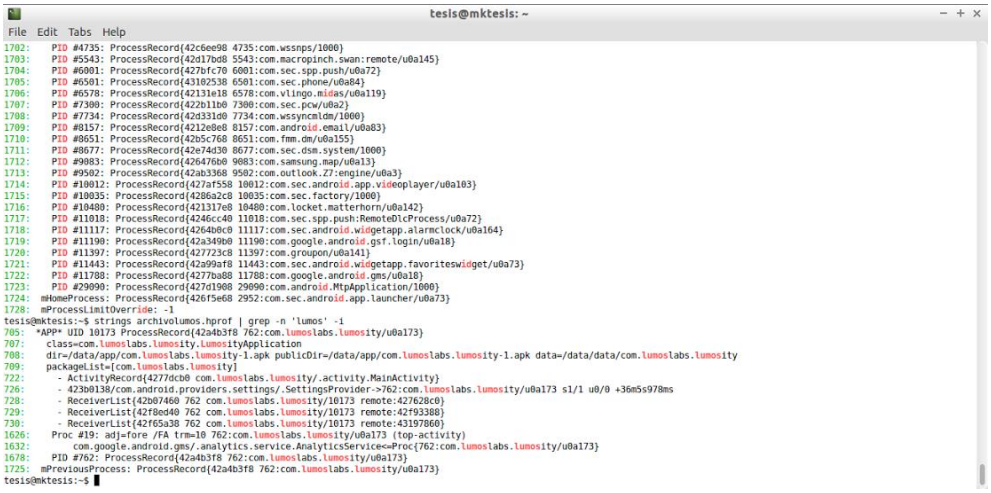
Se realizaron comprobaciones de la memoria del dispositivo y no se identifica el almacenamiento de información sensible abierta o una vez cerrada la aplicación.

Figura 119. Lumosity información de la memoria.



Fuente: Los Autores.

Figura 120. Lumosity búsqueda de información en la memoria.



Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin encontrarse información sensible.

Figura 121. Lumosity búsqueda de información en el archivo hprof.

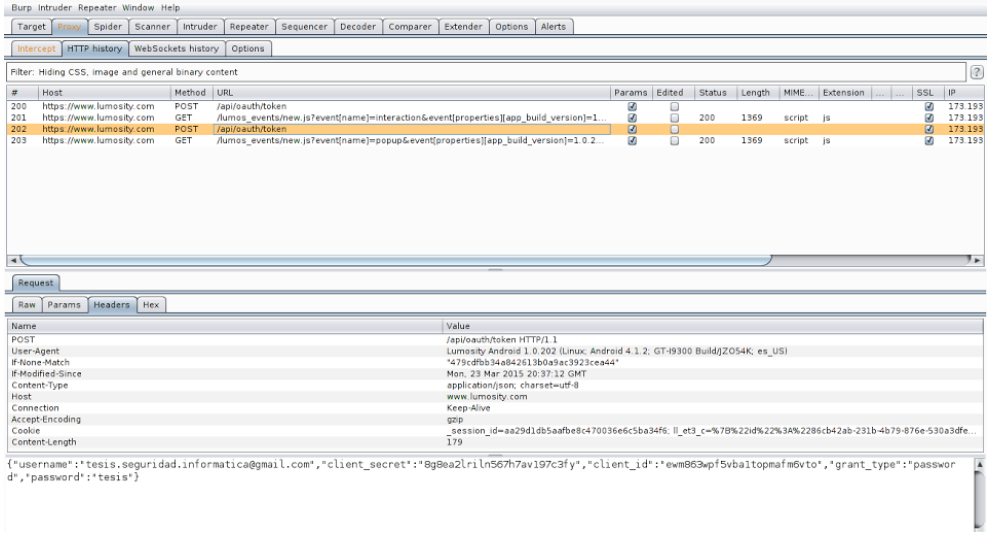
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

1- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Realizada la interceptación del tráfico HTTPS y SSL de la red se encontró que la aplicación al registrarse envía información sensible como el usuario y contraseña de acceso solicitado sin cifrar, convirtiéndose en una vulnerabilidad en las comunicaciones, ya que un atacante puede usar los datos y suplantar la identidad del usuario.

Figura 122. Lumosity información de autenticación.

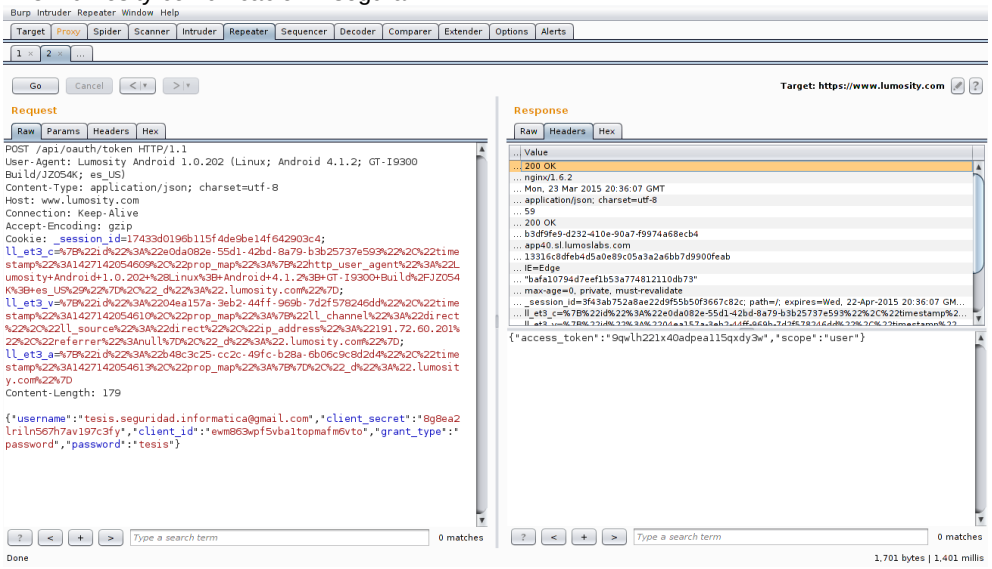


Fuente: Los Autores.

2- Determinar si se usan protocolos de comunicación de forma segura

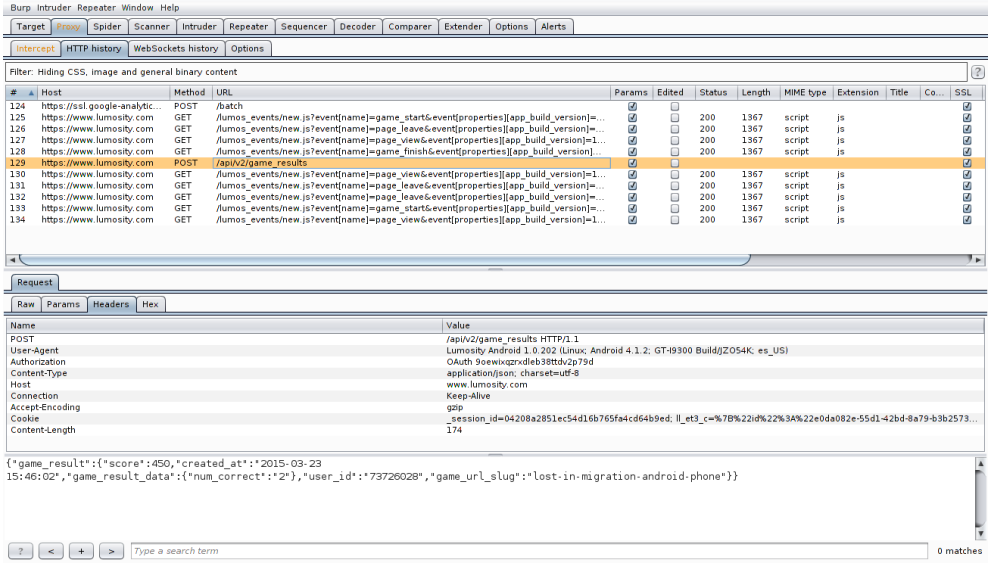
Se observa que los protocolos de comunicación certificados SSL y protocolos HTTPS. Pero al enviar información sensible como el usuario y contraseña sin cifrar del acceso a la aplicación, se convierte en una vulnerabilidad alta.

Figura 123. Lumosity comunicación insegura.



Fuente: Los Autores.

Figura 124. Lumosity comunicación insegura.



Fuente: Los Autores.

V. Aplicación Wish

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
Wish (com.contextlogic.wish)

2- Funcionalidad básica
Es una aplicación que permite encontrar y adquirir productos (muebles, joyas, ropa, cosmético, etc.); adicionalmente se puede crear listas de favoritos, obtener información del vendedor y las opiniones de los compradores.

3- ¿La aplicación realiza transacciones electrónicas?

[X] Si [ ] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[X] Si [ ] No

Figura 125. Wish Permisos

Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

[ ]	NFC	[ ]	Bluetooth
[ ]	GPS	[X]	Cámara
[ ]	Micrófono	[ ]	Sensores
[X]	USB		

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

	Telefonía (SMS, teléfono)	X	Contactos
	Recepción de datos de aplicaciones y otros servicios en el dispositivo		Google Wallet
X	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)		Correo electrónico
	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 126. Wish interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[X] Si [ ] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[ ] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

B- Análisis estático

General

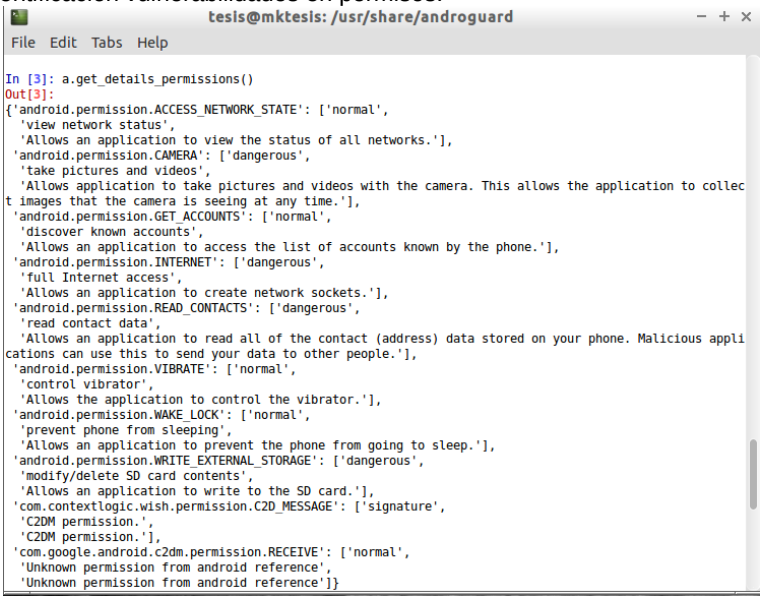
1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.
El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- CAMERA permite tomar imágenes y videos con la cámara, lo que conlleva a que una aplicación puede revisar las imágenes de la cámara y enviarlas.
- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.
- READ_CONTACTS permite el acceso a la información de contactos del dispositivo, sin poder controlar que destino se le dará a estos datos.
- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.

Figura 127. Wish revisión de permisos.

Fuente: Los Autores.

Figura 128. Wish identificación vulnerabilidades en permisos.

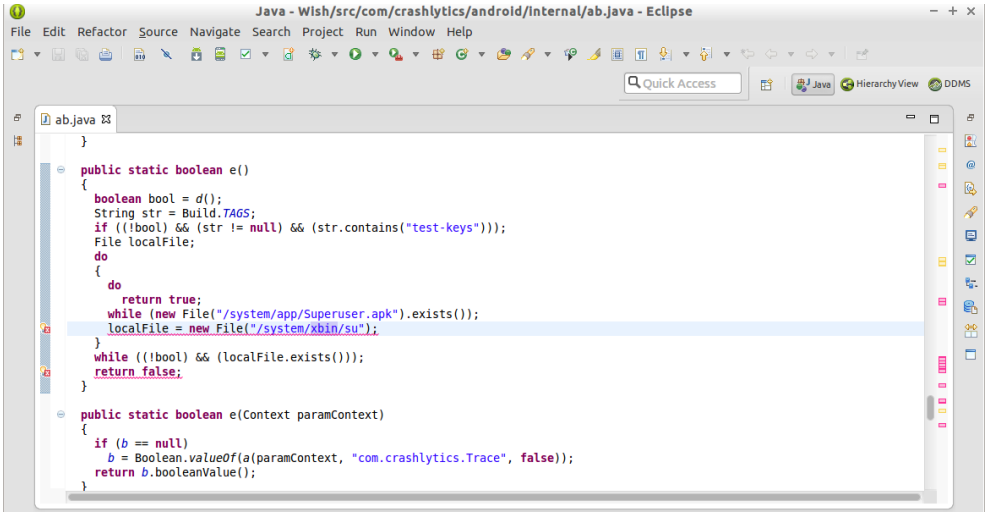


Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

Si, valida la existencia de la aplicación “Superuser.apk” para determinar si el dispositivo esta rooteado.

Figura 129. Wish validación root.



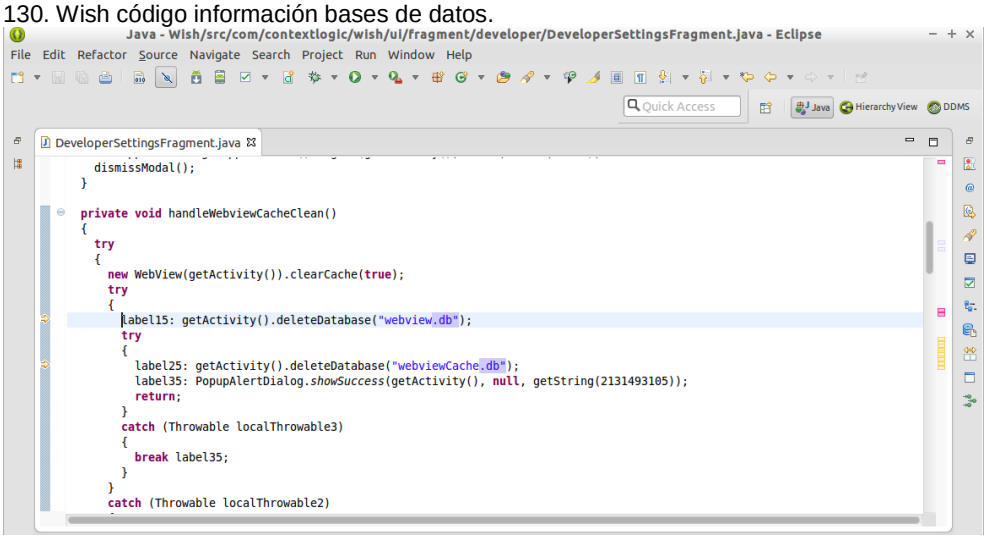
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa una base de datos llamado *webview.db y webviewcache.db*, el archivo que muestra esta información es: *com/contextlogic/wish/ul/fragment/Developer/DeveloperSettingsFragment.java*

Figura 130. Wish código información bases de datos.



Fuente: Los Autores.

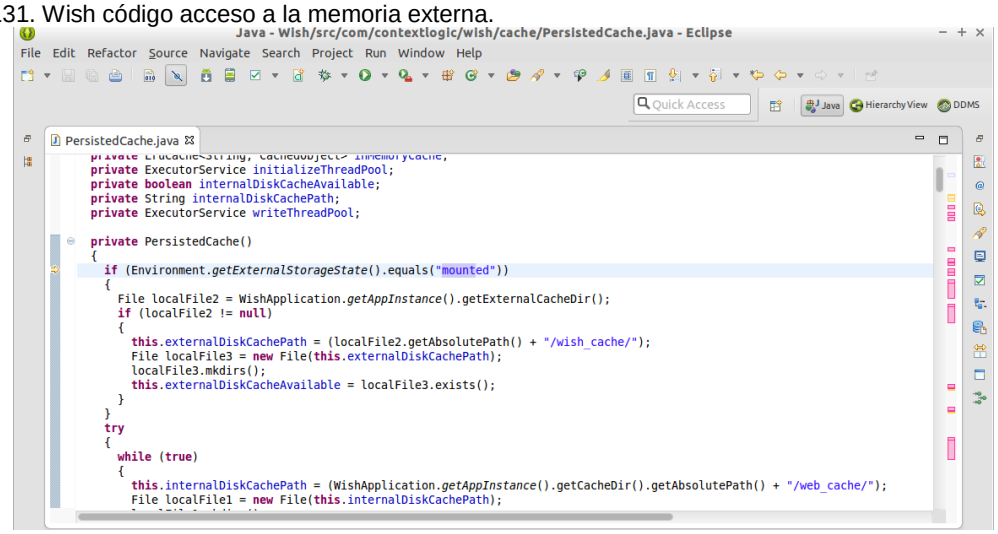
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 131. Wish código acceso a la memoria externa.

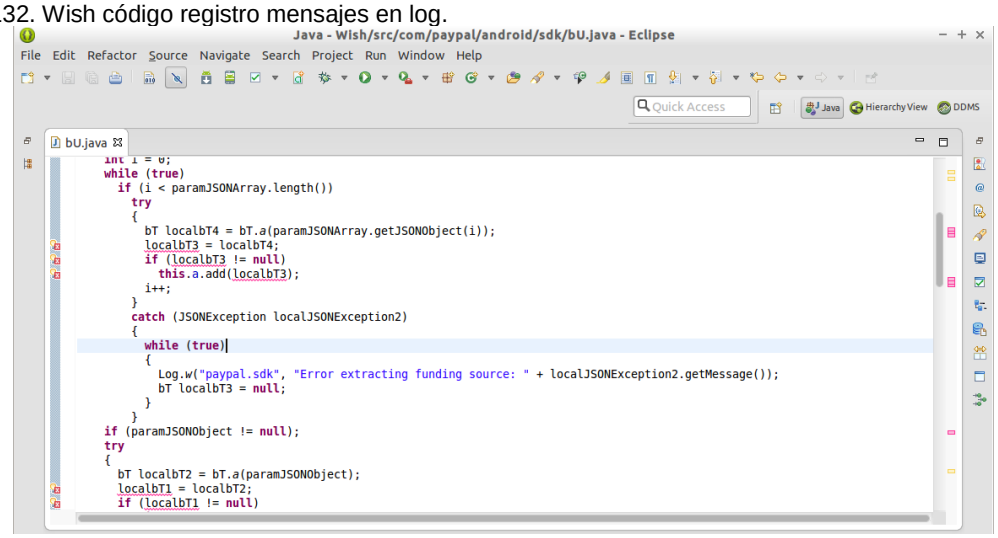


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 132. Wish código registro mensajes en log.



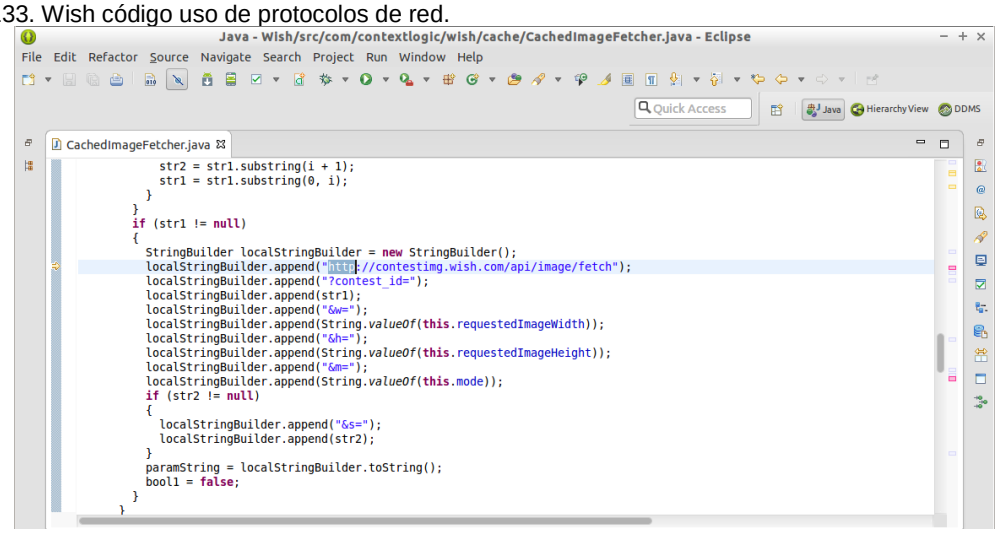
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 133. Wish código uso de protocolos de red.

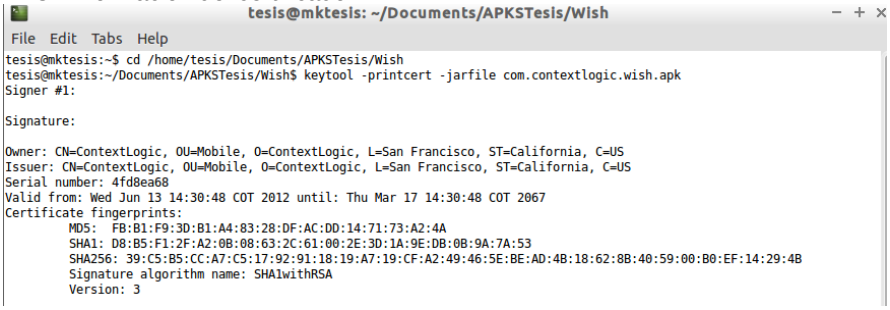


Fuente: Los Autores.

7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

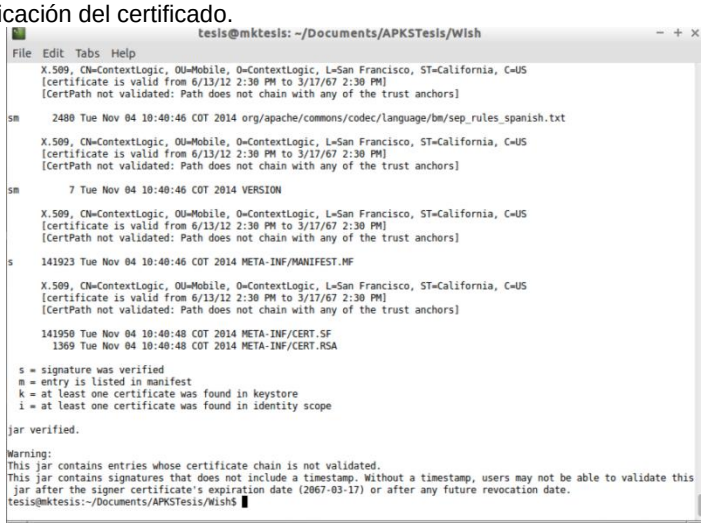
Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 134. Wish información del certificado.



Fuente: Los Autores.

Figura 135. Wish verificación del certificado.



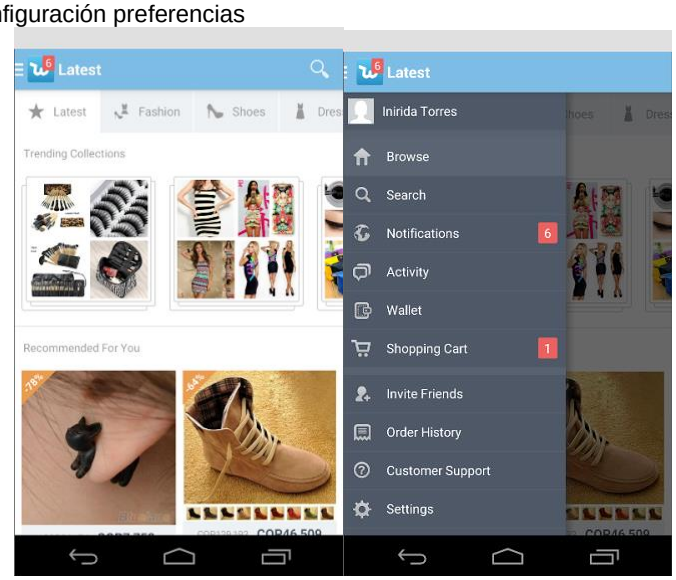
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 136. Wish configuración preferencias



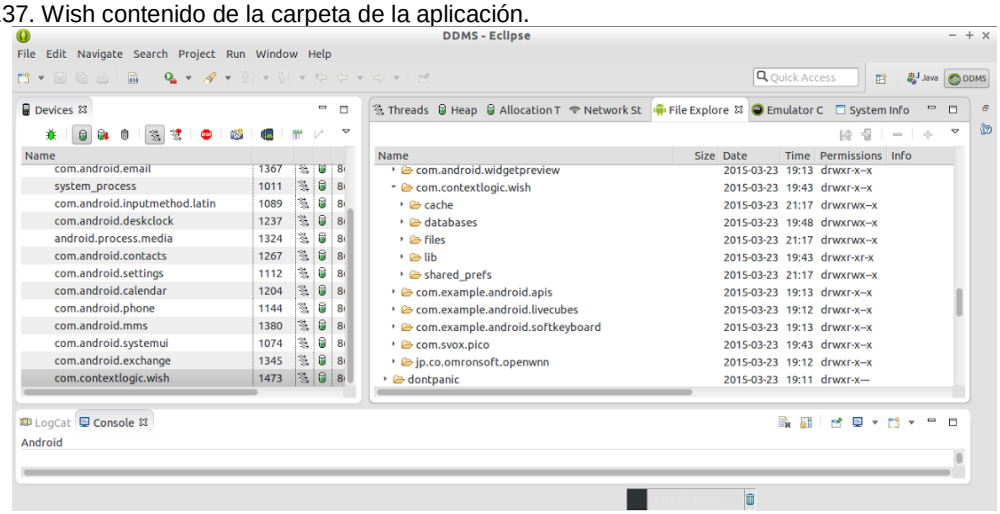
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.contextlogic.wish” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

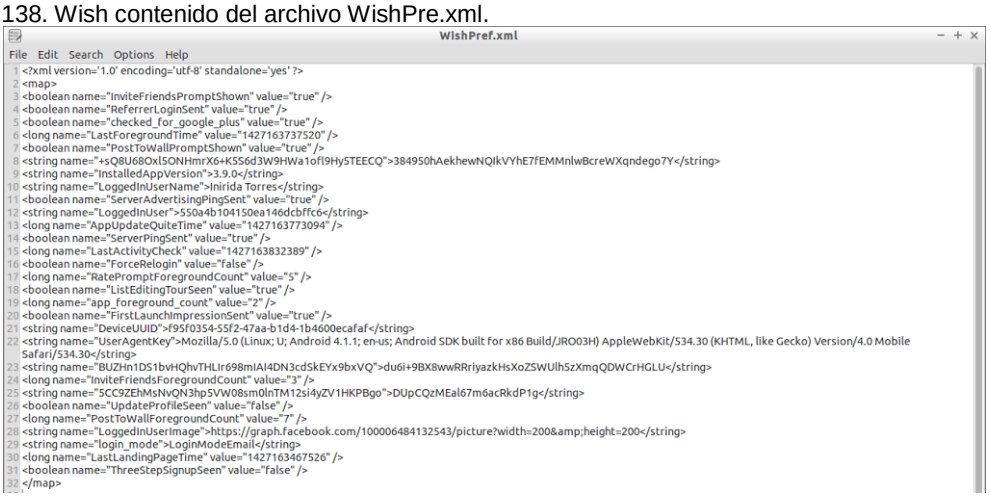
Figura 137. Wish contenido de la carpeta de la aplicación.



Fuente: Los Autores.

Se observa en la carpeta “/shared_prefs”, donde se revisa los archivos “xml” encontrándose almacenamiento de información sensible como es el nombre del usuario de la cuenta.

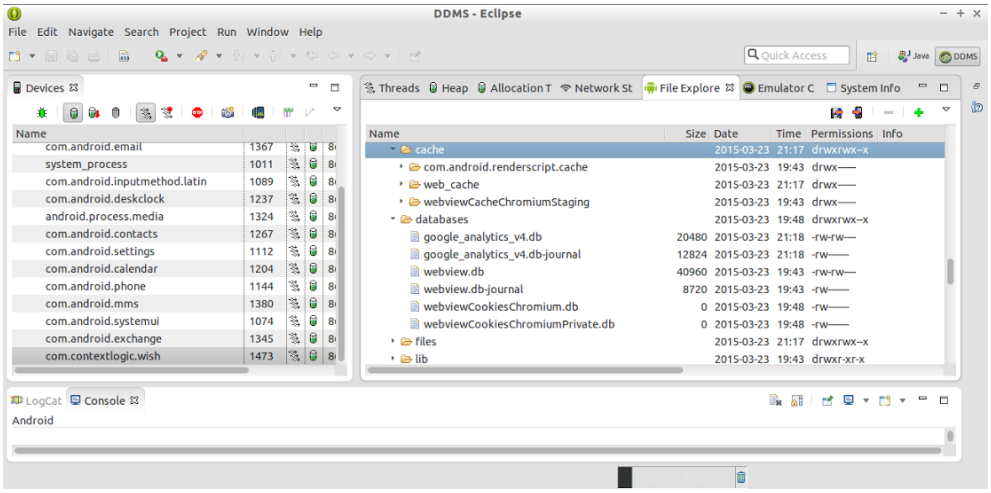
Figura 138. Wish contenido del archivo WishPre.xml.



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “webview.db”, “webviewCookiesChromium.db” y “google_analytics_v4.db”, las cuales son usadas por las API y no contienen información sensible de la aplicación.

Figura 139. Wish ubicación archivo base de datos

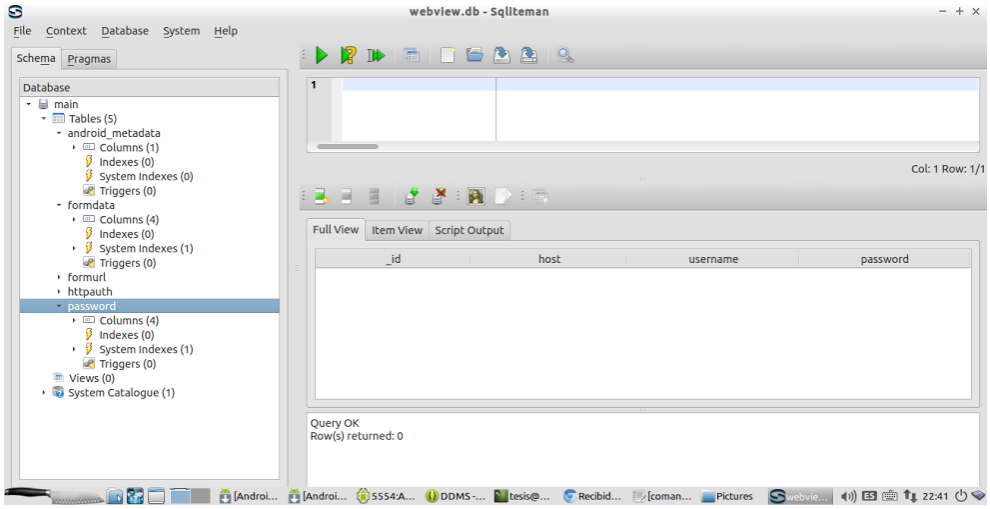


Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos se observa que son bases de datos de las APIs implementadas en la aplicación, las cuales no almacenan información sensible

Figura 140. Wish contenido de la base de datos “webview.db”.



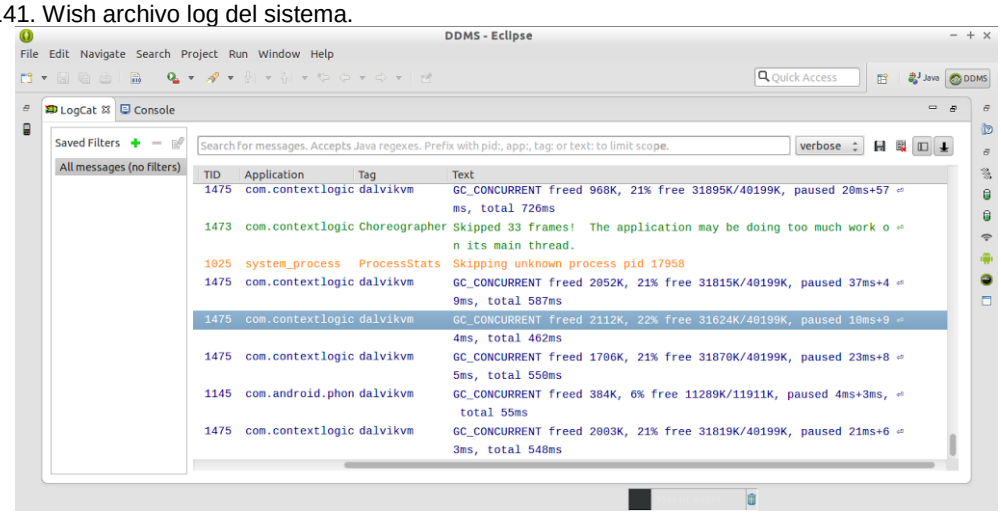
Fuente: Los Autores.

En la carpeta files no se encuentran archivos que manejen información sensible.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log no se observa el almacenamiento de información sensible, solo muestra información de las actividades generadas con cada API.

Figura 141. Wish archivo log del sistema.

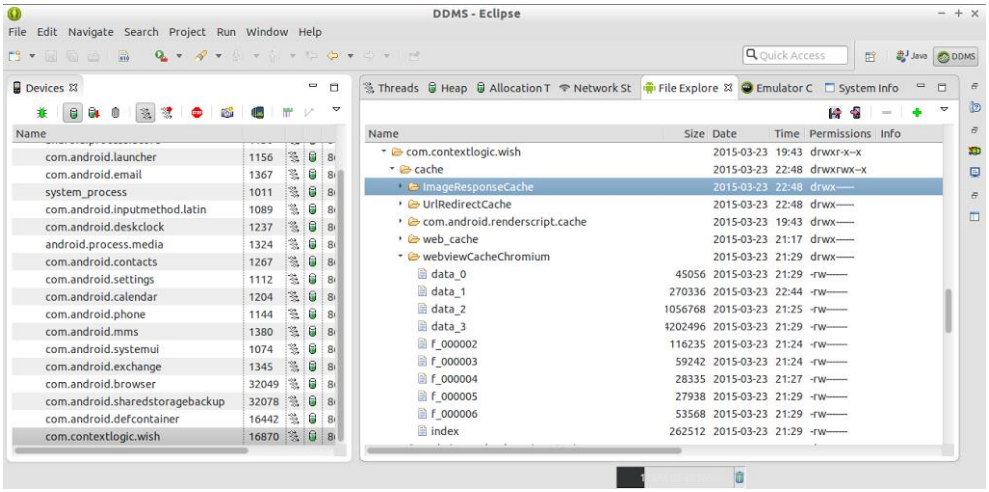


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación en donde no se encontró almacenamiento de información sensible.

Figura 142. Wish contenido de la cache.

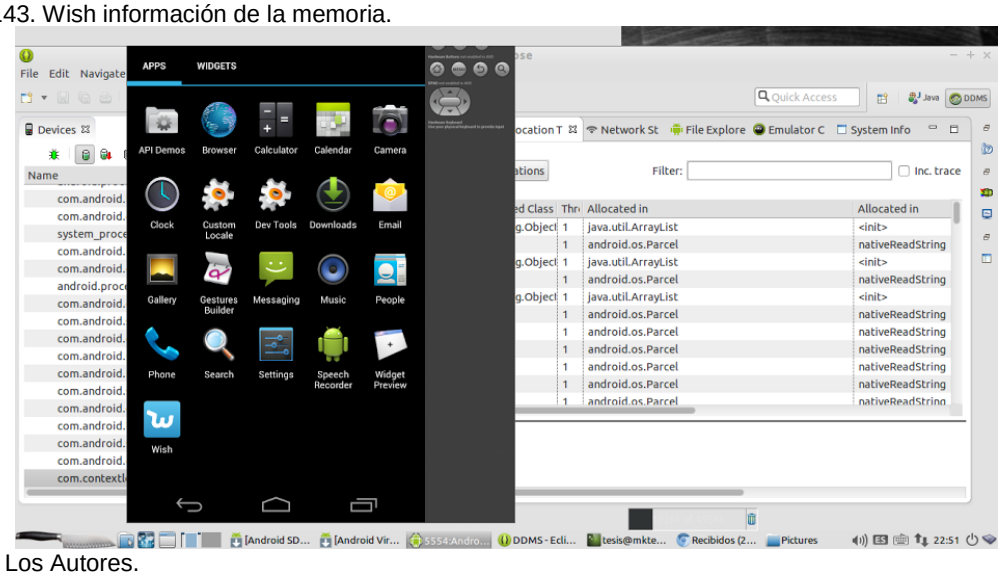


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información sensible de la misma.

Figura 143. Wish información de la memoria.



Fuente: Los Autores.

Figura 144. Wish búsqueda de información en la memoria.

Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin identificar información sensible.

Figura 145. Wish búsqueda de información en el archivo hprof.

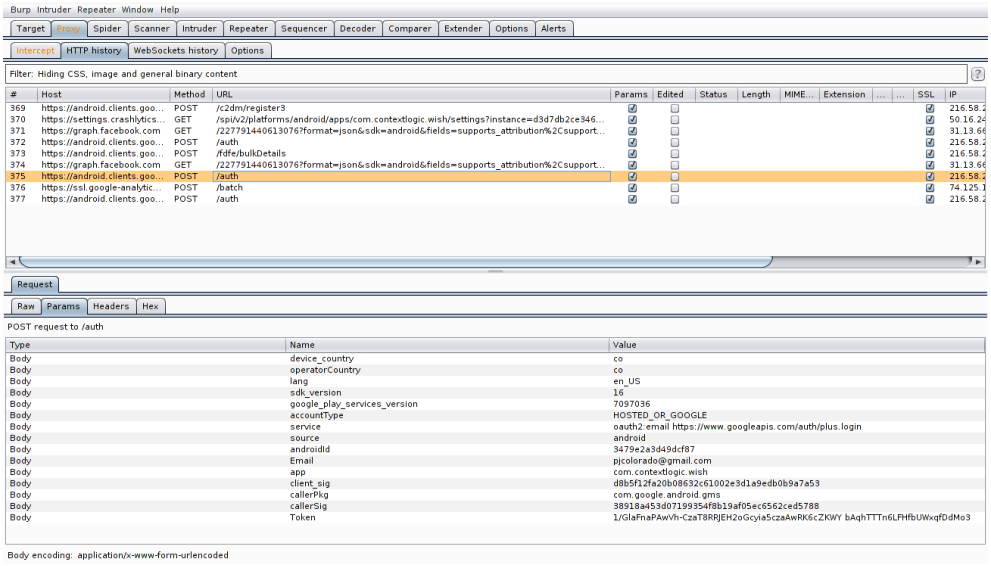
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Realizada la interceptación del tráfico HTTPS y SSL de la red se encontró que la aplicación envía información sensible en la autenticación como es el usuario representado por el correo electrónico sin cifrar, convirtiéndose en una vulnerabilidad en las comunicaciones.

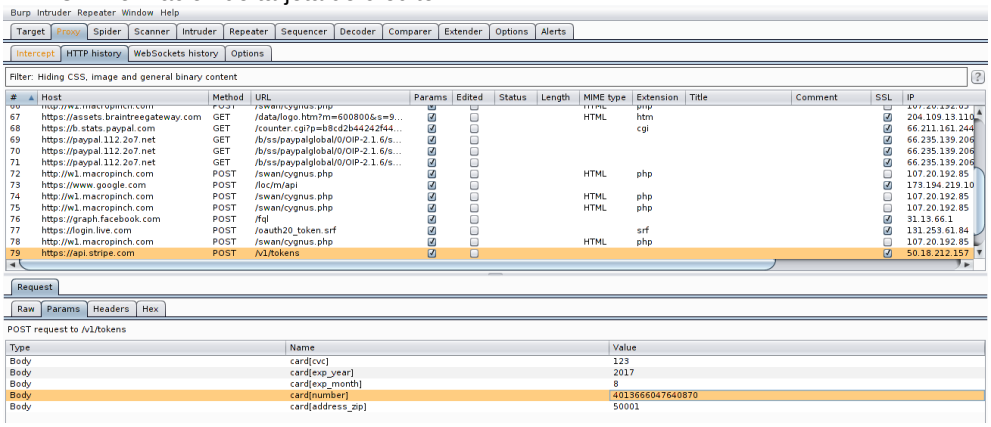
Figura 146. Wish información de autenticación.



Fuente: Los Autores.

Se revisa la información de la tarjeta de crédito en una compra identificando el envío de información sensible como es los datos de la tarjeta crédito sin cifrar, convirtiéndose en una vulnerabilidad en las comunicaciones.

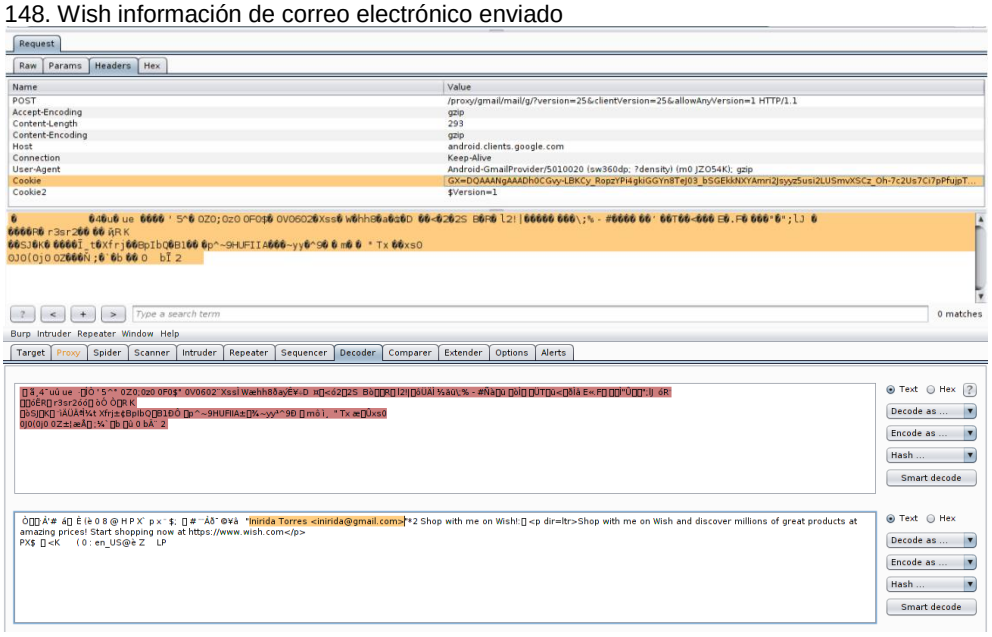
Figura 147. Wish información de tarjeta de crédito



Fuente: Los Autores.

Se revisa el envío de un mensaje de correo electrónico a través de la aplicación identificando el envío de información sensible como la dirección de correo electrónico del destinatario y el contenido del mensaje convirtiéndose en una vulnerabilidad en las comunicaciones.

Figura 148. Wish información de correo electrónico enviado

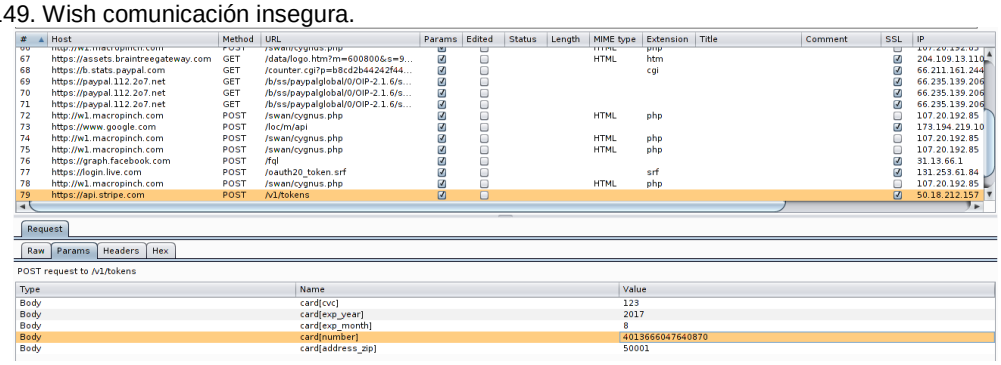


Fuente: Los Autores

9- Determinar si se usan protocolos de comunicación de forma segura.

Se observa que los protocolos de comunicación no se usan de forma segura, la aplicación admite comunicación a través de los protocolos HTTPS y SSL, pero envía información sensible como son los datos de la tarjeta de crédito.

Figura 149. Wish comunicación insegura.



Fuente: Los Autores.

VI. Aplicación Shazam

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
Shazam (com.shazam.android)

2- Funcionalidad básica
Aplicación que permite identificar música y programas de televisión, permitiendo escuchar fragmentos de canciones, compartir en redes sociales y realizar compras en Amazon o Google play.

3- ¿La aplicación realiza transacciones electrónicas?

[X] Si [ ] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[X] Si [ ] No

Figura 150. Shazam Permisos

Fuente: Los Autores.

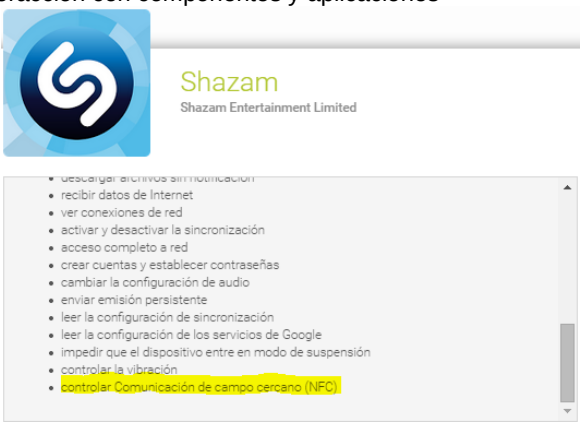
4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

X	NFC		Bluetooth
	GPS	X	Cámara
	Micrófono		Sensores
X	USB		

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

	Telefonía (SMS, teléfono)	X	Contactos
	Recepción de datos de aplicaciones y otros servicios en el dispositivo		Google Wallet
X	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)		Correo electrónico
	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 151. Shazam interacción con componentes y aplicaciones



Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

X	Si		No

7- Identificar las interfaces de red inalámbrica utilizadas:

	Wi-Fi (802.11)	X	NFC		Bluetooth

B- Análisis estático

General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

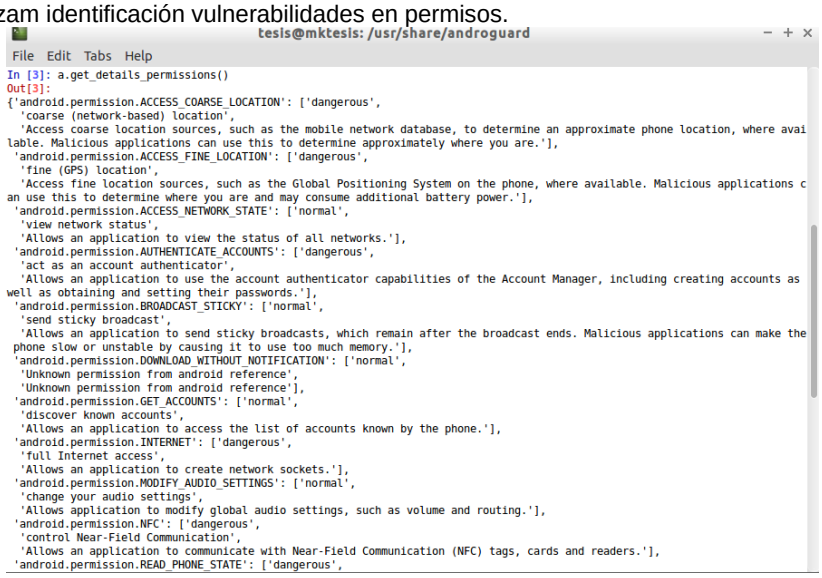
El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- ACCESS_COARSE_LOCATION permite que una aplicación acceda a la ubicación aproximada derivado de las fuentes de ubicación de red, como las redes telefonía móvil y Wi - Fi.
- ACCESS_FINE_LOCATION permite que una aplicación acceda a la ubicación precisa de las fuentes de ubicación, como el GPS, las redes de telefonía móvil y Wi - Fi.
- AUTHENTICATE_ACCOUNTS permite que una aplicación pueda actuar como administrador de cuentas para la autenticación
- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.
- NFC permite realizar operaciones de entrada /salida sobre NFC.
- READ_PHONE_STATE permite acceso de sólo lectura al estado del teléfono.
- RECORD_AUDIO permite que una aplicación grabe audio.
- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.

Figura 152. Shazam revisión de permisos.

Fuente: Los Autores.

Figura 153. Shazam identificación vulnerabilidades en permisos.

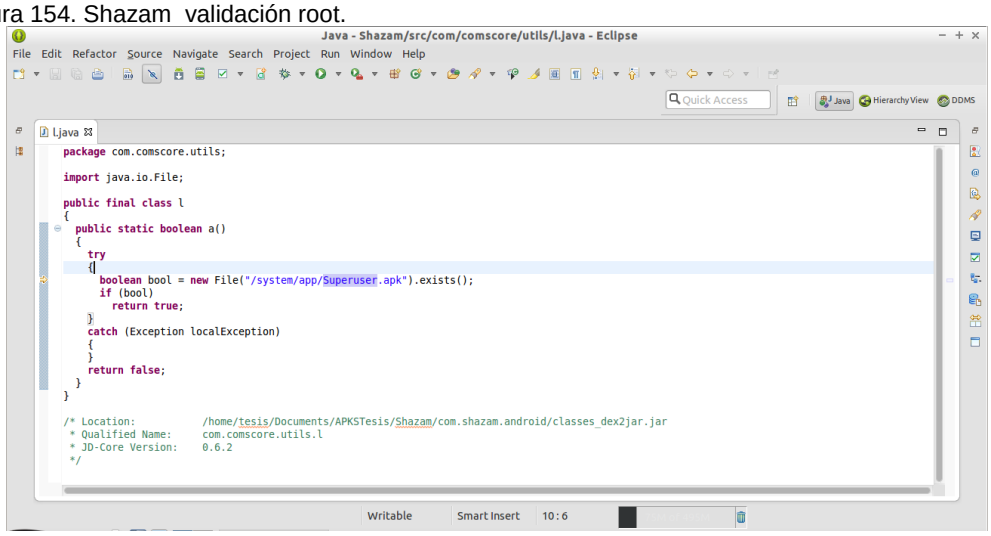


Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

Si, valida la existencia de la aplicación “Superuser.apk” para determinar si el dispositivo esta rooteado.

Figura 154. Shazam validación root.



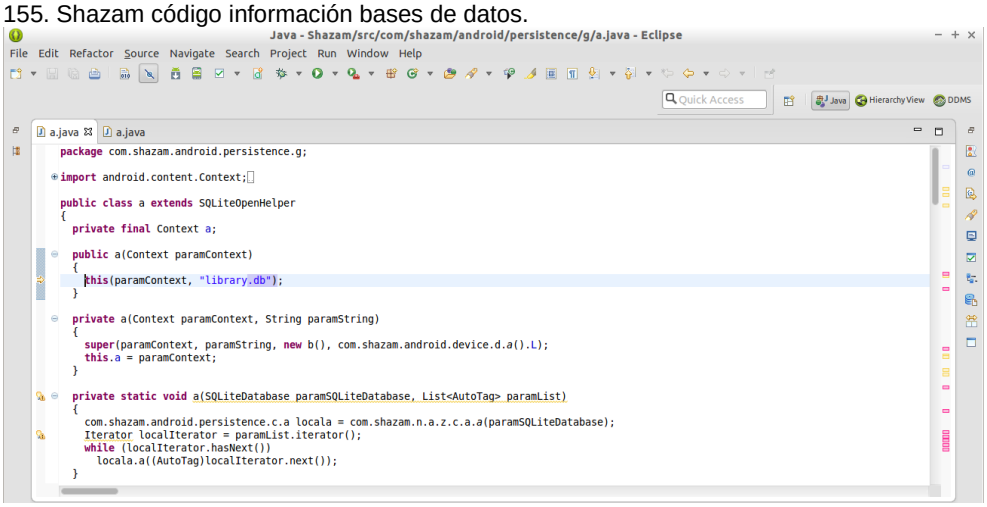
Fuente: Los Autores.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa una base de datos llamado *Library.db*, el archivo que muestra esta información es: *com/shazam/android/persistence/g/a-java.java*

Figura 155. Shazam código información bases de datos.



Fuente: Los Autores.

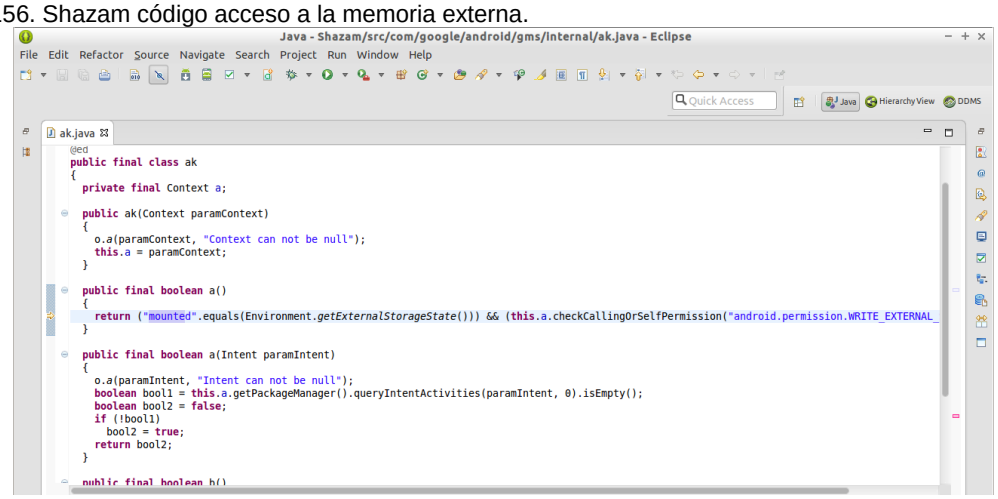
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 156. Shazam código acceso a la memoria externa.

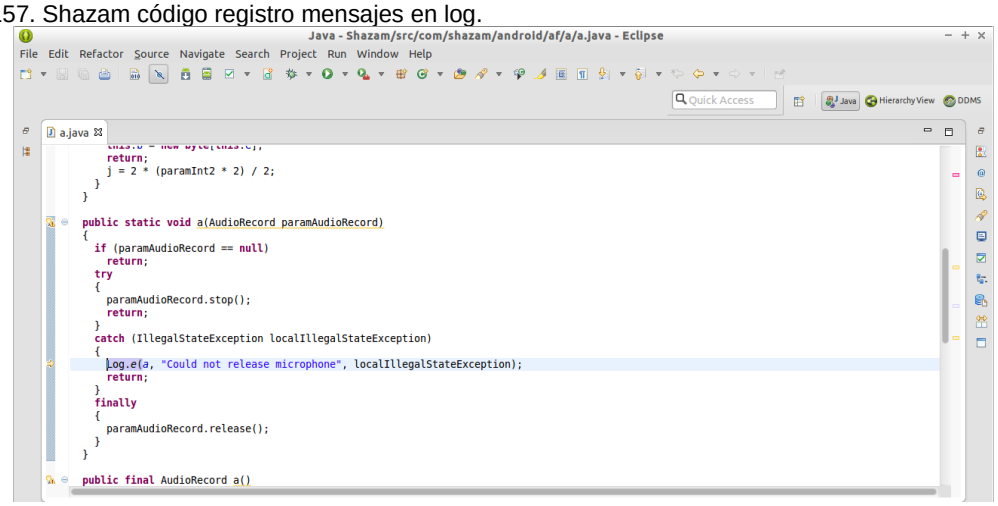


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 157. Shazam código registro mensajes en log.



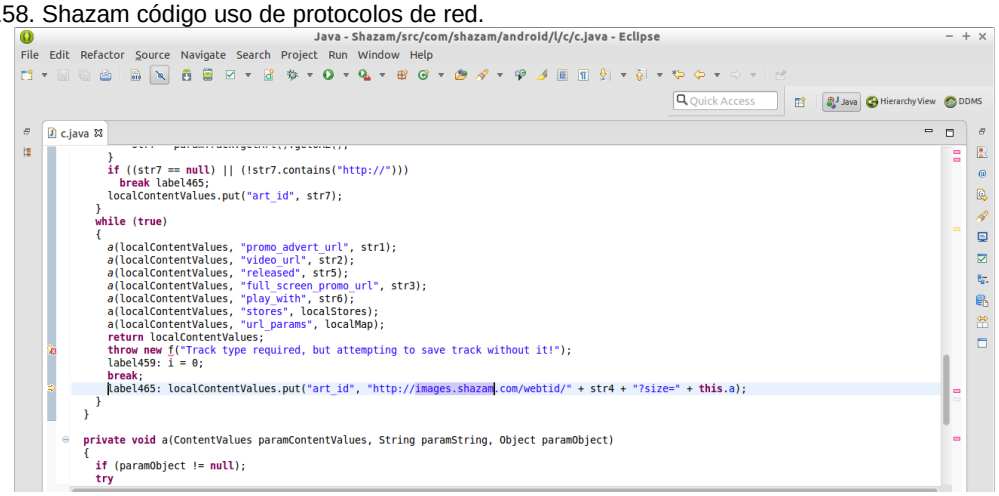
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 158. Shazam código uso de protocolos de red.

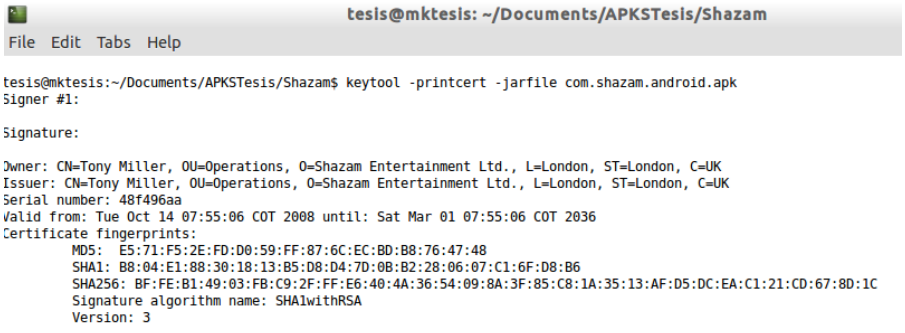


Fuente: Los Autores.

7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

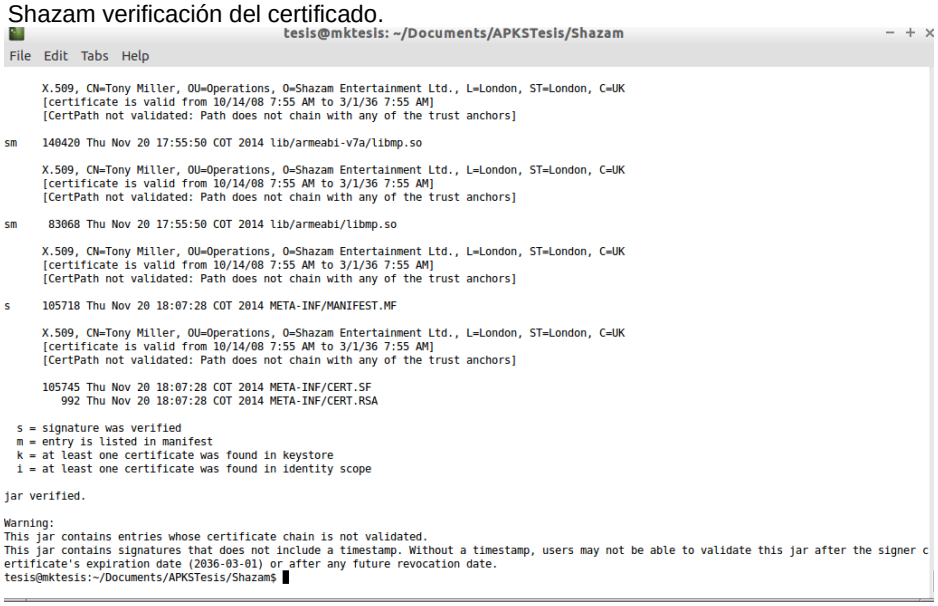
Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 159. Shazam información del certificado.



Fuente: Los Autores.

Figura 160. Shazam verificación del certificado.



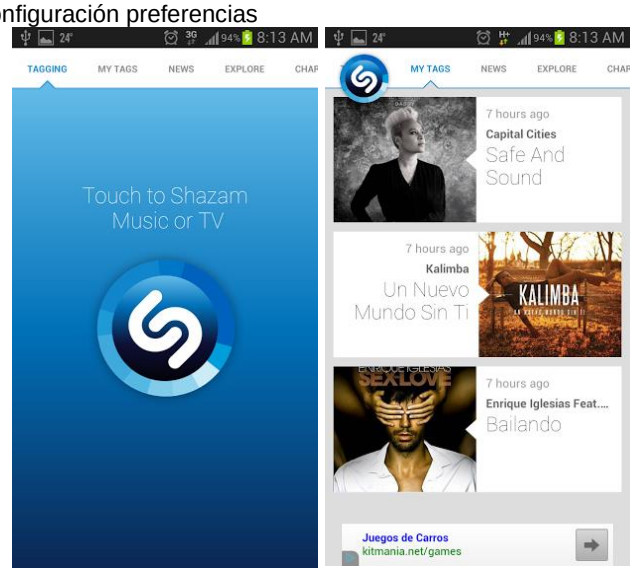
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 161. Shazam configuración preferencias



Fuente: Los Autores.

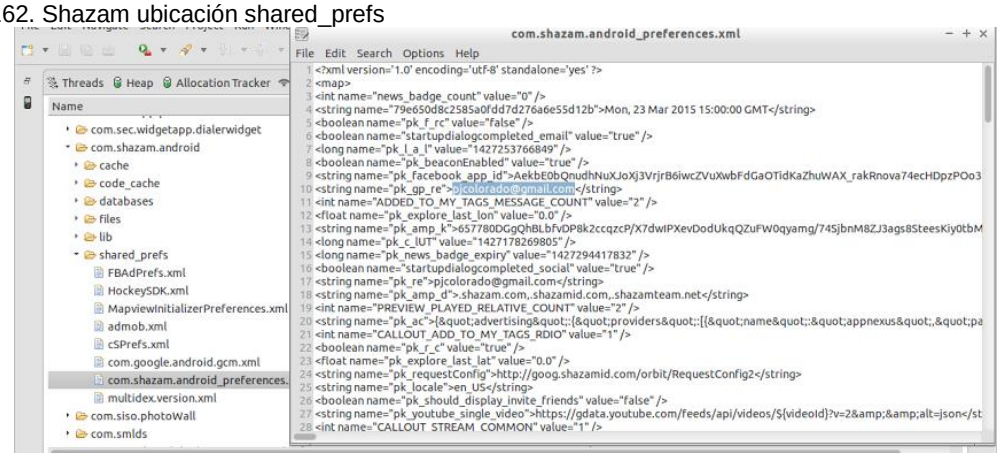
M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.shazam.android” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

Se observa en la carpeta “/shared_prefs”, donde se revisa los archivos “xml” encontrandose almacenamiento de información sensible como el correo electrónico del usuario con el cual inicia sesión en la aplicación.

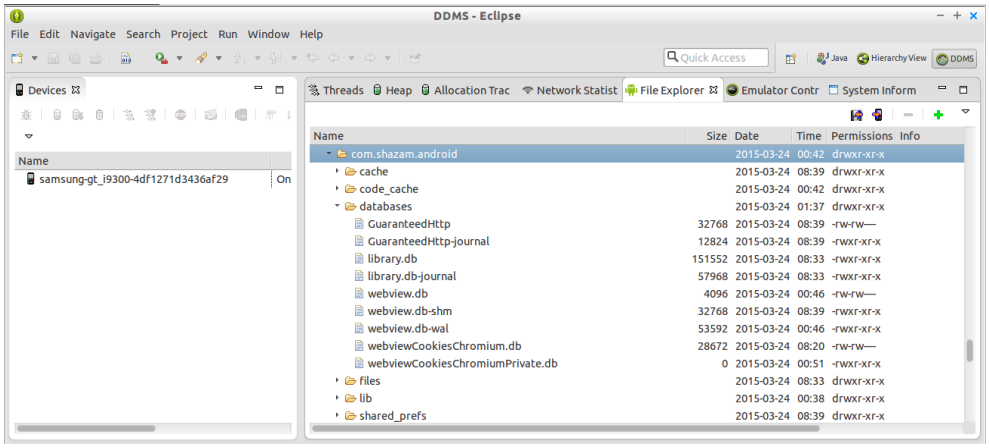
Figura 162. Shazam ubicación shared_prefs



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “library.db”, “library.db-journal”, “webviewCookiesChromium.db” y “webview.db”, las cuales son usadas por las API y no contienen información sensible de la aplicación.

Figura 163. Shazam ubicación archivo base de datos



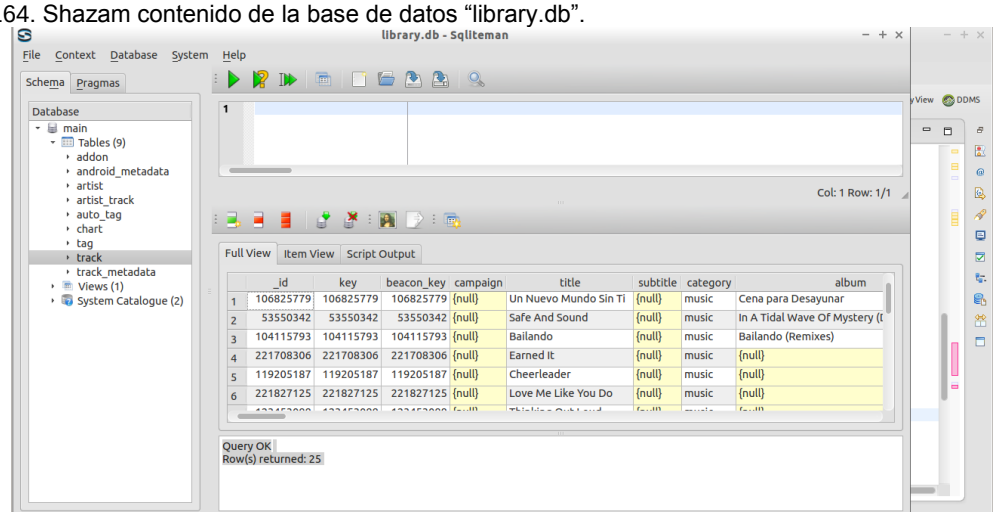
Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “library.db” se muestra en la tabla *artista* muestra el listado de los artistas de las canciones que han sido shazameadas y en la tabla *track* el listado de canciones con sus datos de identificación.

La información encontrada no se considera sensible por cuanto no se observa almacenamiento de datos del usuario.

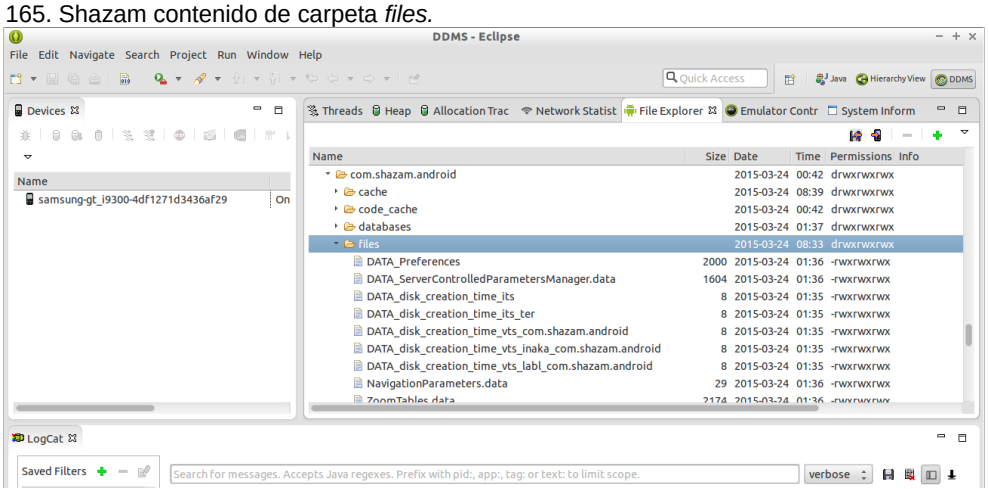
Figura 164. Shazam contenido de la base de datos “library.db”.



Fuente: Los Autores.

En la carpeta files no se encuentran archivos que manejen información sensible.

Figura 165. Shazam contenido de carpeta *files.*

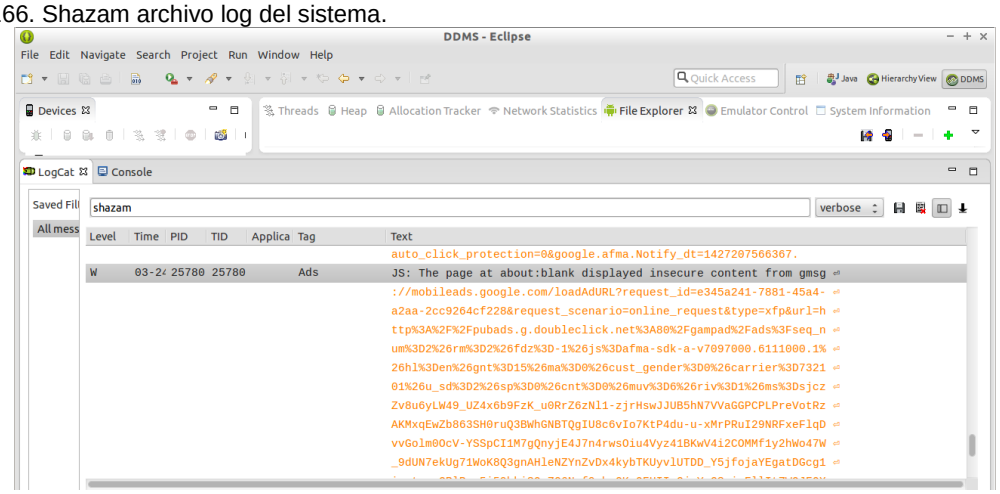


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log se observa el almacenamiento de información cifrada, pero no es posible ver su contenido

Figura 166. Shazam archivo log del sistema.

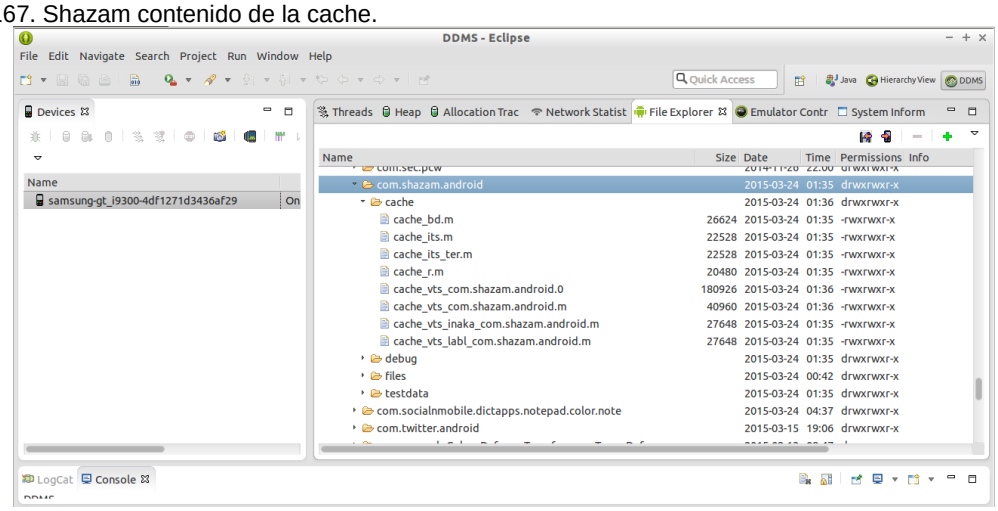


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

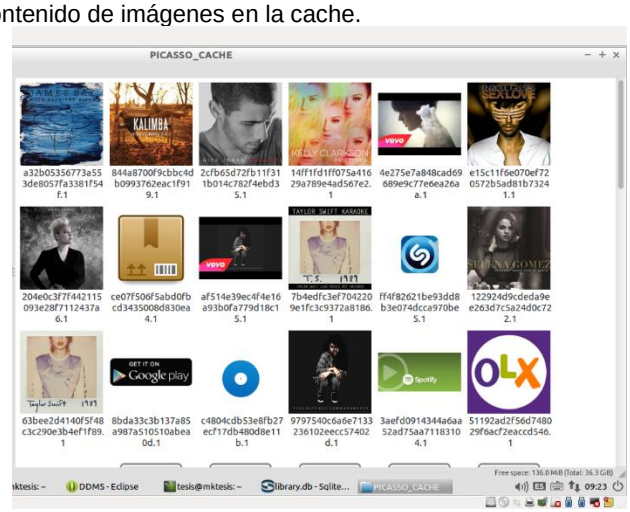
Se procedió a revisar la carpeta cache de la aplicación en donde no se encontró almacenamiento de información sensible.

Figura 167. Shazam contenido de la cache.



Fuente: Los Autores.

Figura 168. Shazam contenido de imágenes en la cache.

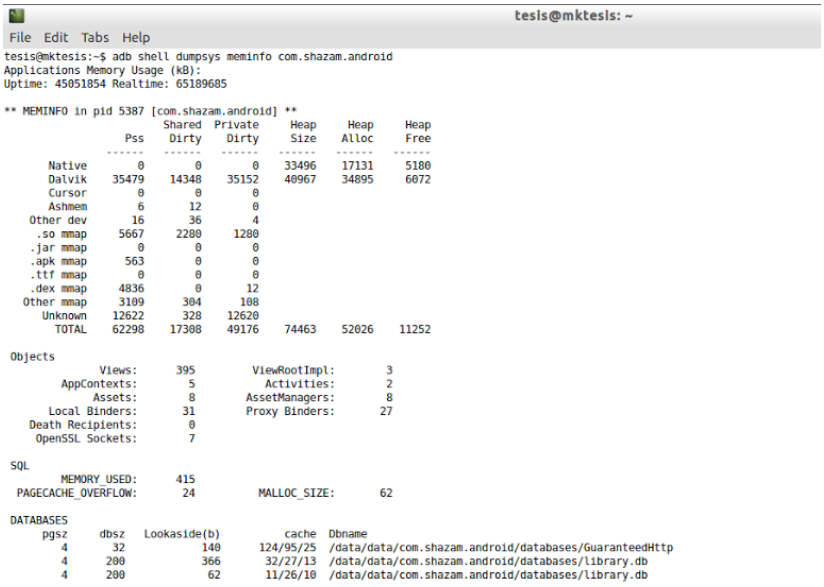


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información sensible de la misma.

Figura 169. Shazam información de la memoria.



```
tesis@mktesis: ~
File  Edit  Tabs  Help
tesis@mktesis:~$ adb shell dumpsys meminfo com.shazam.android
Applications Memory Usage (kB):
Uptime: 45051854 Realtime: 65189685

** MEMINFO in pid 5387 [com.shazam.android] **
                   Pss  Private  Private  Heap  Heap  Heap
                   Pss    Dirty    Dirty  Size  Alloc  Free
                 ------  ------  ------  ------  ------  ------
       Native        0       0       0   33496   17131    5180
       Dalvik    35479   14348   35152   40967   34895    6072
       Cursor        0       0       0
       Ashmem        6      12       0
    Other dev       16      36       4
     .so mmap     5667    2280    1280
    .jar mmap        0       0       0
    .apk mmap      563       0       0
    .ttf mmap        0       0       0
    .dex mmap     4836       0      12
   Other mmap     3109     304     108
      Unknown    12622     328   12620
        TOTAL    62298   17308   49176   74463   52026   11252

 Objects
               Views:      395         ViewRootImpl:        3
         AppContexts:        5           Activities:        2
              Assets:        8       AssetManagers:        8
      Local Binders:       31       Proxy Binders:       27
   Death Recipients:        0
    OpenSSL Sockets:        7

 SQL
         MEMORY_USED:      415
  PAGECACHE_OVERFLOW:       24          MALLOC_SIZE:       62

 DATABASES
      pgsz     dbsz   Lookaside(b)          cache  Dbname
         4       32            140      124/95/25  /data/data/com.shazam.android/databases/GuaranteedHttp
         4      200            366       32/27/13  /data/data/com.shazam.android/databases/library.db
         4      200             62       11/26/10  /data/data/com.shazam.android/databases/library.db
```

Fuente: Los Autores.

Figura 170. Shazam búsqueda de información en la memoria.

```
tesis@mktesis: ~
File  Edit  Tabs  Help
1614:    user #0 uid=1000
tesis@mktesis:~$ strings shazam.hprof | grep -n 'shazam' -i
331:  *APP* UID 10014 ProcessRecord{42af3338 5387:com.shazam.android/u0a14}
333:    class=com.shazam.android.ShazamApplication
334:    dir=/data/app/com.shazam.android-1.apk publicDir=/data/app/com.shazam.android-1.apk data=/data/data/com.shazam.android
335:    packageList=[com.shazam.android]
348:      - ActivityRecord{42cba3c8 com.shazam.android/.activities.MainActivity}
349:      - ActivityRecord{42d515a0 com.shazam.android/.activities.MusicDetailsActivity}
351:      - ServiceRecord{42e8e330 com.shazam.android/.service.player.MusicPlayerService}
353:      - ConnectionRecord{42f32068 com.shazam.android/.service.player.MusicPlayerService:@42e8f5d0}
354:      - ConnectionRecord{430f9d00 com.shazam.android/.service.player.MusicPlayerService:@42e06ed0}
355:      - ConnectionRecord{42a90710 com.shazam.android/.service.player.MusicPlayerService:@42a6ff18}
356:      - ConnectionRecord{42a9f448 com.shazam.android/.service.player.MusicPlayerService:@42a49ac8}
358:      - com.shazam.android.persistence.providers.NewsSummarySyncStubContentProvider
359:        -> ContentProviderRecord{42e87bc0 com.shazam.android/.persistence.providers.NewsSummarySyncStubContentProvider}
360:      - com.shazam.android.persistence.providers.ImportTagsFreeContentProvider
361:        -> ContentProviderRecord{42e706e8 com.shazam.android/.persistence.providers.ImportTagsFreeContentProvider}
362:      - com.shazam.android.persistence.providers.TrackSyncStubContentProvider
363:        -> ContentProviderRecord{42e49b20 com.shazam.android/.persistence.providers.TrackSyncStubContentProvider}
364:      - com.shazam.android.service.guaranteedhttp.GuaranteedHttpProvider
365:        -> ContentProviderRecord{42e87c78 com.shazam.android/.service.guaranteedhttp.GuaranteedHttpProvider}
366:      - com.shazam.android.persistence.LibraryDAO
367:        -> ContentProviderRecord{42e49a40 com.shazam.android/.persistence.LibraryDAO}
369:      - 423b0138/com.android.providers.settings/.SettingsProvider->5387:com.shazam.android/u0a14 s1/1 u0/0 +7m30s701ms
371:      - ReceiverList{4318edc8 5387 com.shazam.android/10014 remote:43a9ed78}
372:      - ReceiverList{42c89c60 5387 com.shazam.android/10014 remote:42c89b38}
373:      - ReceiverList{431371a0 5387 com.shazam.android/10014 remote:43416b68}
374:      - ReceiverList{4276dea8 5387 com.shazam.android/10014 remote:42eaa258}
375:      - ReceiverList{42d15418 5387 com.shazam.android/10014 remote:42d12640}
376:      - ReceiverList{42b40080 5387 com.shazam.android/10014 remote:426ef348}
377:      - ReceiverList{42f7a500 5387 com.shazam.android/10014 remote:430a2018}
378:      - ReceiverList{42a819c0 5387 com.shazam.android/10014 remote:42a982d8}
379:      - ReceiverList{430af268 5387 com.shazam.android/10014 remote:430ecd00}
380:      - ReceiverList{430eada8 5387 com.shazam.android/10014 remote:43180940}
381:      - ReceiverList{42a80960 5387 com.shazam.android/10014 remote:42ad30a0}
382:      - ReceiverList{42a57128 5387 com.shazam.android/10014 remote:42b6f0c0}
1640:    Proc #20: adj=fore /FA trm=15 5387:com.shazam.android/u0a14 (top-activity)
1719:    PID #5387: ProcessRecord{42af3338 5387:com.shazam.android/u0a14}
tesis@mktesis:~$
```

Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin identificar información sensible.

Figura 171. Shazam búsqueda de información en el archivo hprof.

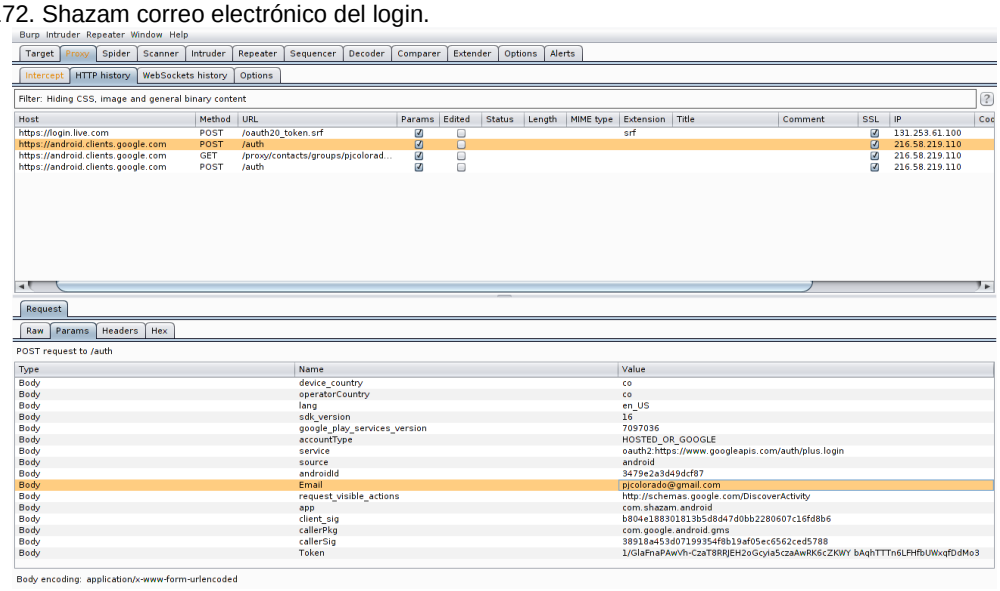
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Se realiza el loggeo en la aplicación logrando interceptarse el correo electrónico con el cual el usuario inicia sesión identificándose la transmisión de información sensible.

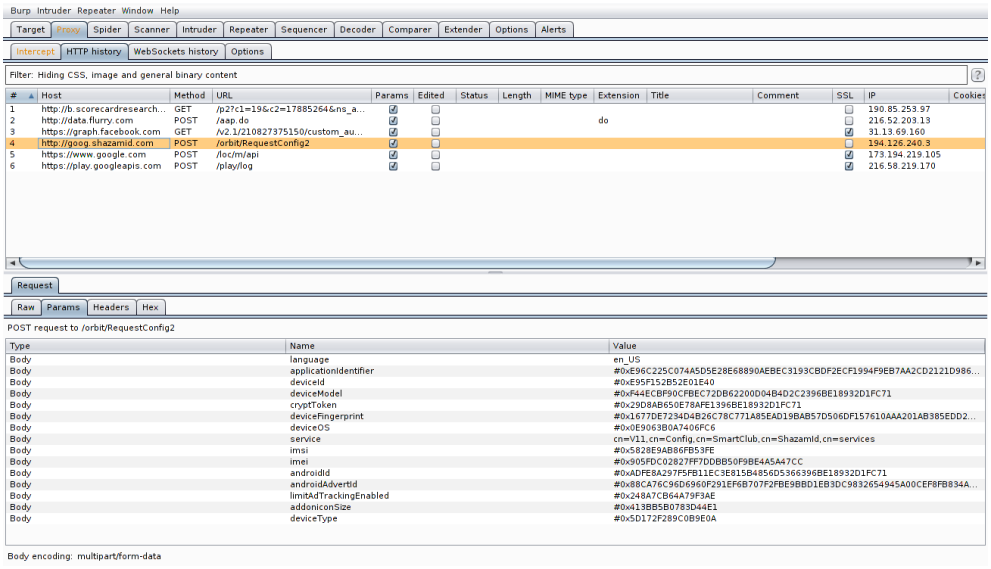
Figura 172. Shazam correo electrónico del login.



Fuente: Los Autores.

Realizada la interceptación del tráfico de la red se encontró que la aplicación utiliza protocolos HTTP y SSL en donde cifra la información sensible transmitida.

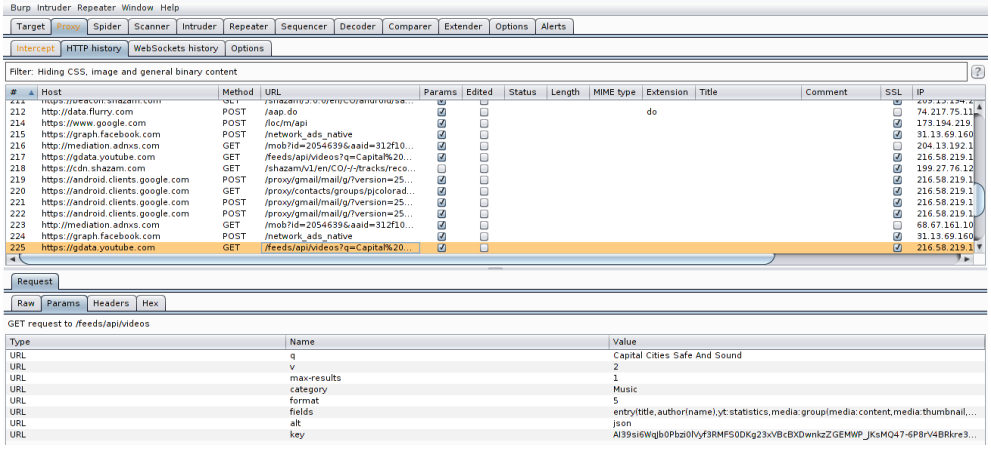
Figura 173. Shazam información transmitida cifrada.



Fuente: Los Autores.

Se revisa la recepción de la información de las canciones shazameadas en donde puede identificarse solamente el nombre de la canción.

Figura 174. Shazam información canción encontrada.

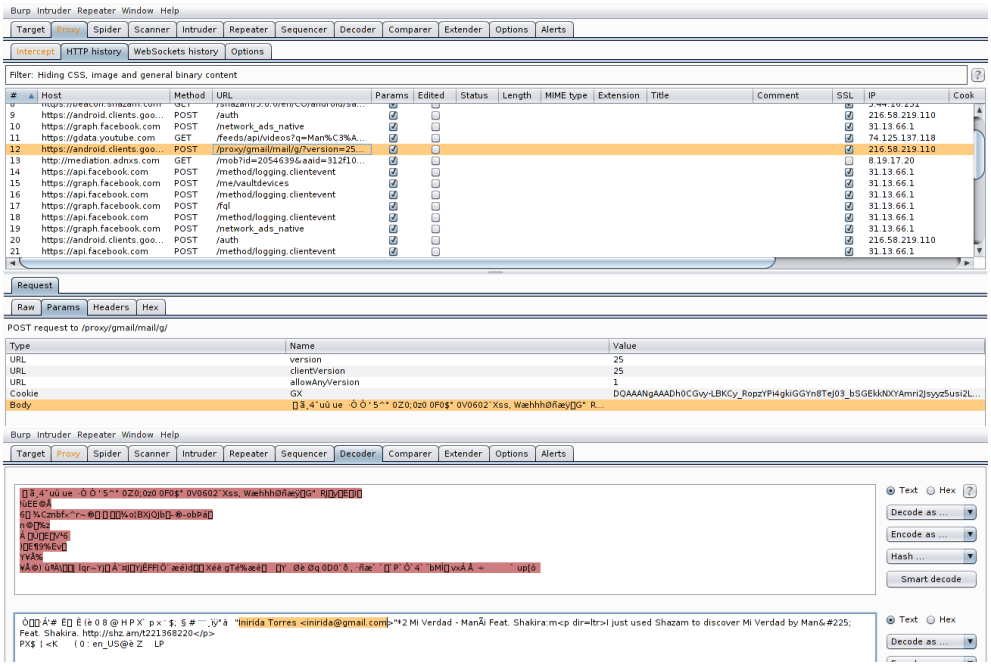


Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

Realizando un envío de un mensaje texto a través de la aplicación se obtiene los datos del correo electrónico y la canción a compartir, comprometiendo la información sensible.

Figura 175. Shazam información correo decodificado



Fuente: Los Autores.

VII. Aplicación IF by IFTTT

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
IF by IFTTT (com.ifttt.ifttt)

2- Funcionalidad básica
Permite crear conexiones potentes con una sencilla premisa “si ocurre esto, haz aquello” con canales (como, por ejemplo, Facebook, Dropbox y Gmail, así como dispositivos como el Termostato Nest, Fitbit y Hue de Phillips).

3- ¿La aplicación realiza transacciones electrónicas?

[ ] Si [X] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[ ] Si [ ] No

Figura 176. IFTTT Permisos

Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

	NFC		Bluetooth
X	GPS	X	Cámara
	Micrófono	X	Sensores
X	USB		

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

X	Telefonía (SMS, teléfono)	X	Contactos
	Recepción de datos de aplicaciones y otros servicios en el dispositivo		Google Wallet
	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)		Correo electrónico
	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 177. IFTTT interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[X] Si [ ] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[X] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

B- Análisis estático

General

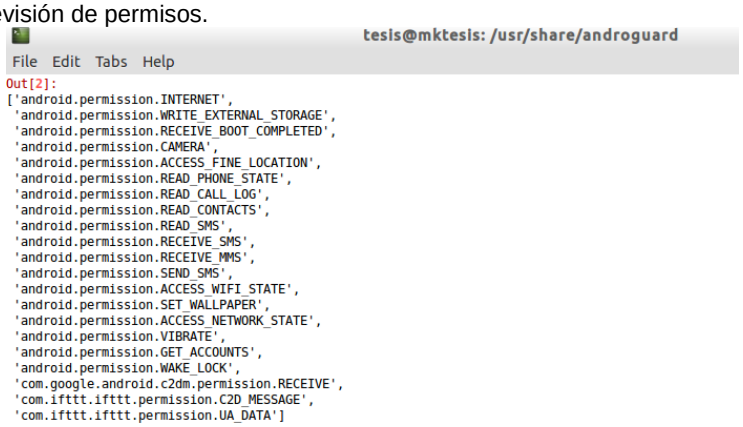
1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- ACCESS_FINE_LOCATION permite que una aplicación acceda a la ubicación precisa de las fuentes de ubicación, como el GPS, las redes de telefonía móvil y Wi - Fi.
- CAMERA permite acceder a la cámara del dispositivo.
- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.
- READ_CALL_LOG permite leer el registro de llamadas del usuario.
- READ_CONTACTS permite leer los datos de los contactos del usuario.
- READ_PHONE_STATE permite acceso de sólo lectura al estado del teléfono.
- READ_SMS permite leer los mensajes SMS.

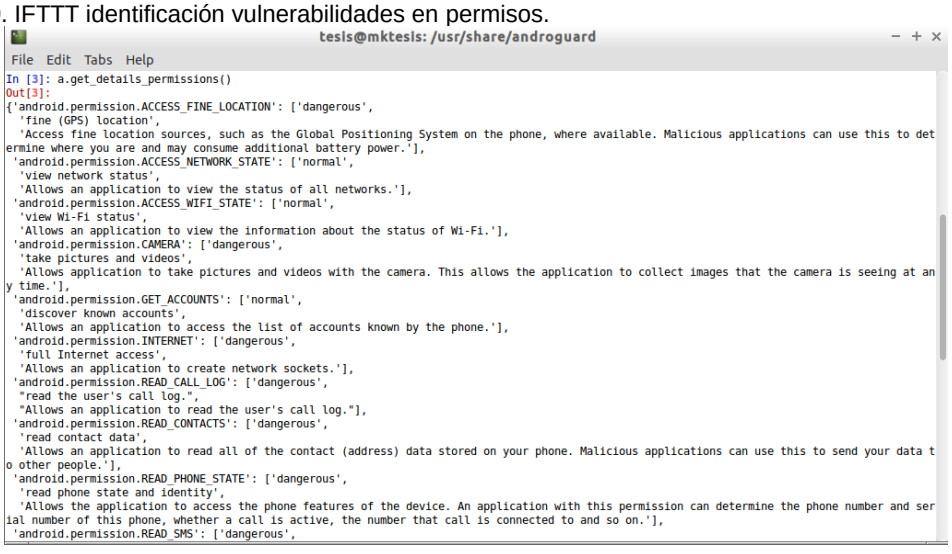
- RECEIVE_MMS permite que una aplicación pueda monitorizar los mensajes MMS entrantes, para grabar o realizar el procesamiento en ellos.
- RECEIVE SMS permite que una aplicación pueda monitorizar los mensajes SMS entrantes, para grabar o realizar el procesamiento en ellos.
- SEND_SMS permite que una aplicación envíe mensajes SMS.
- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.

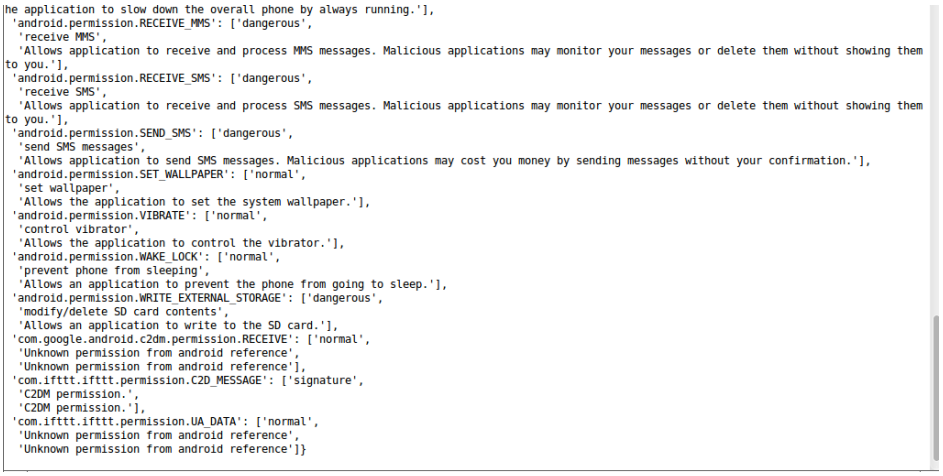
Figura 178. IFTTT revisión de permisos.



Fuente: Los Autores.

Figura 179. IFTTT identificación vulnerabilidades en permisos.





```
he application to slow down the overall phone by always running.'],
 'android.permission.RECEIVE_MMS': ['dangerous',
  'receive MMS',
  'Allows application to receive and process MMS messages. Malicious applications may monitor your messages or delete them without showing them
to you.'],
 'android.permission.RECEIVE_SMS': ['dangerous',
  'receive SMS',
  'Allows application to receive and process SMS messages. Malicious applications may monitor your messages or delete them without showing them
to you.'],
 'android.permission.SEND_SMS': ['dangerous',
  'send SMS messages',
  'Allows application to send SMS messages. Malicious applications may cost you money by sending messages without your confirmation.'],
 'android.permission.SET_WALLPAPER': ['normal',
  'set wallpaper',
  'Allows the application to set the system wallpaper.'],
 'android.permission.VIBRATE': ['normal',
  'control vibrator',
  'Allows the application to control the vibrator.'],
 'android.permission.WAKE_LOCK': ['normal',
  'prevent phone from sleeping',
  'Allows an application to prevent the phone from going to sleep.'],
 'android.permission.WRITE_EXTERNAL_STORAGE': ['dangerous',
  'modify/delete SD card contents',
  'Allows an application to write to the SD card.'],
 'com.google.android.c2dm.permission.RECEIVE': ['normal',
  'Unknown permission from android reference',
  'Unknown permission from android reference'],
 'com.ifttt.ifttt.permission.C2D_MESSAGE': ['signature',
  'C2DM permission.',
  'C2DM permission.'],
 'com.ifttt.ifttt.permission.UA_DATA': ['normal',
  'Unknown permission from android reference',
  'Unknown permission from android reference']}
```

Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

No. Realizada la revisión del código fuente no se encontró uso de métodos de validación de este parámetro en la búsqueda de instrucciones con los comandos “xbin”, “su”, “sbin”, “system”.

Los dispositivos rooteados no incluyen todas las protecciones de seguridad en el sistema operativo permitiendo el acceso total a información y datos de aplicaciones.

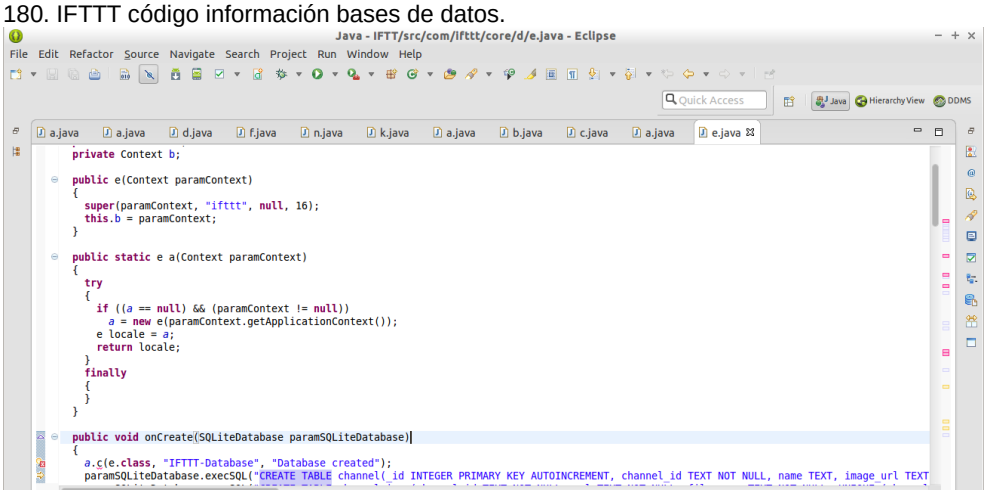
M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa una base de datos llamado *IFTTT-database*, el archivo que muestra esta información es:
com/ifttt/core/d/e.java

Se encontró información de una base de datos del api Urbansirship denominada *ua_analytics.db* y *ua_richpush.db.*

Figura 180. IFTTT código información bases de datos.



Fuente: Los Autores.

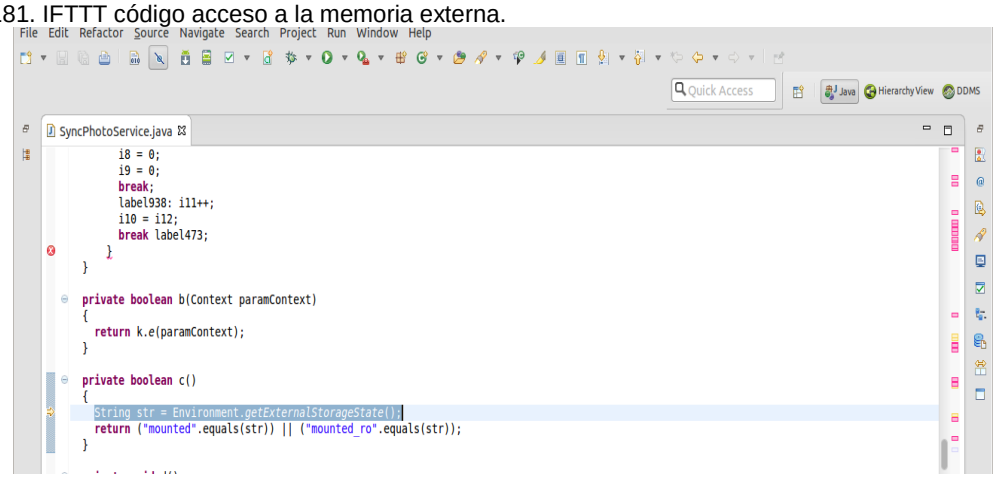
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 181. IFTTT código acceso a la memoria externa.

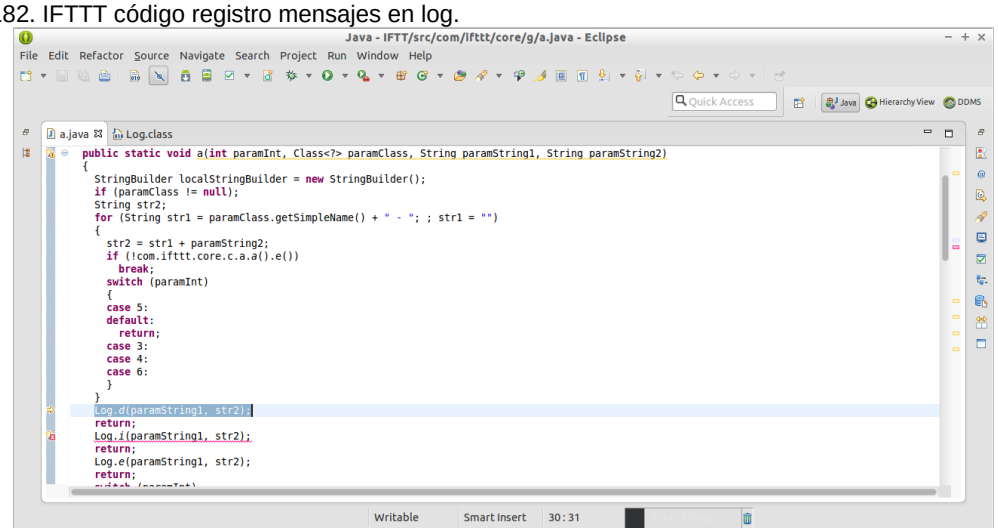


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 182. IFTTT código registro mensajes en log.



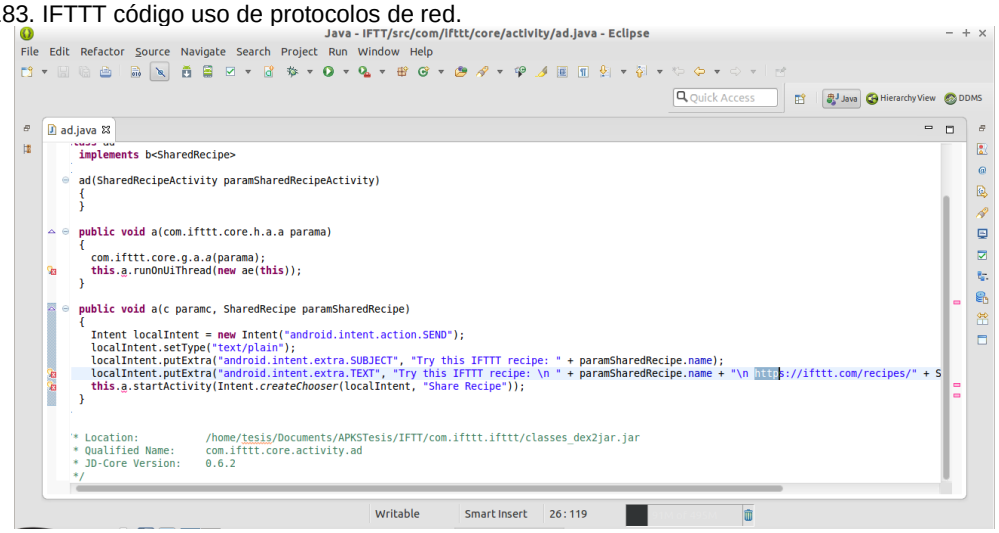
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 183. IFTTT código uso de protocolos de red.



Fuente: Los Autores.

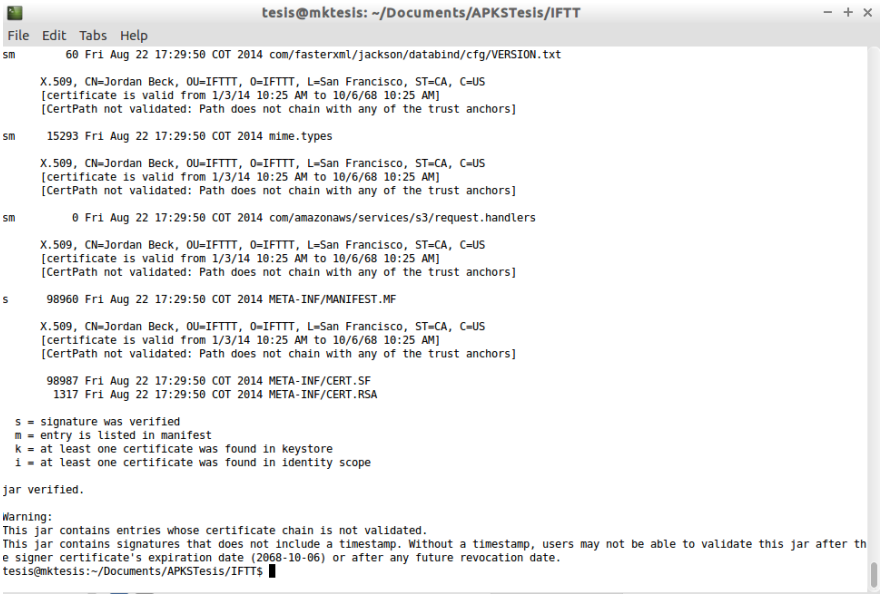
7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 184. IFTTT información del certificado.

Fuente: Los Autores.

Figura 185. IFTTT verificación del certificado.



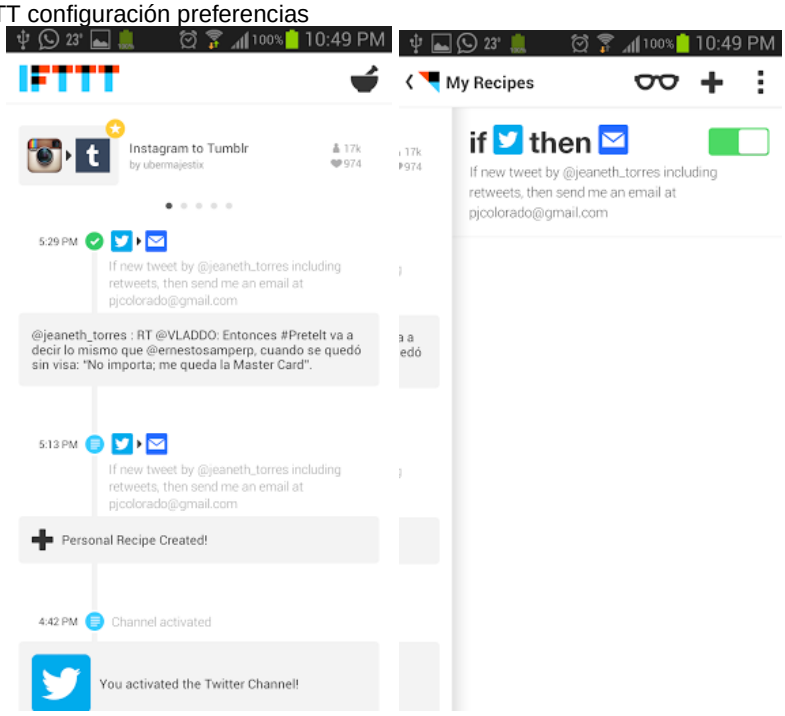
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 186. IFTTT configuración preferencias



Fuente: Los Autores.

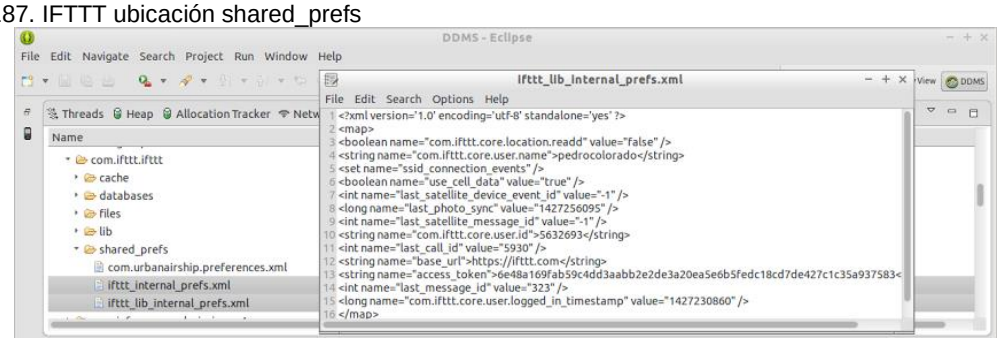
M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.ifttt.ifttt” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

Se observa en la carpeta “/shared_prefs”, el archivo *ifttt_lib_internal_prefs.xml* donde se identifica el nombre de usuario.

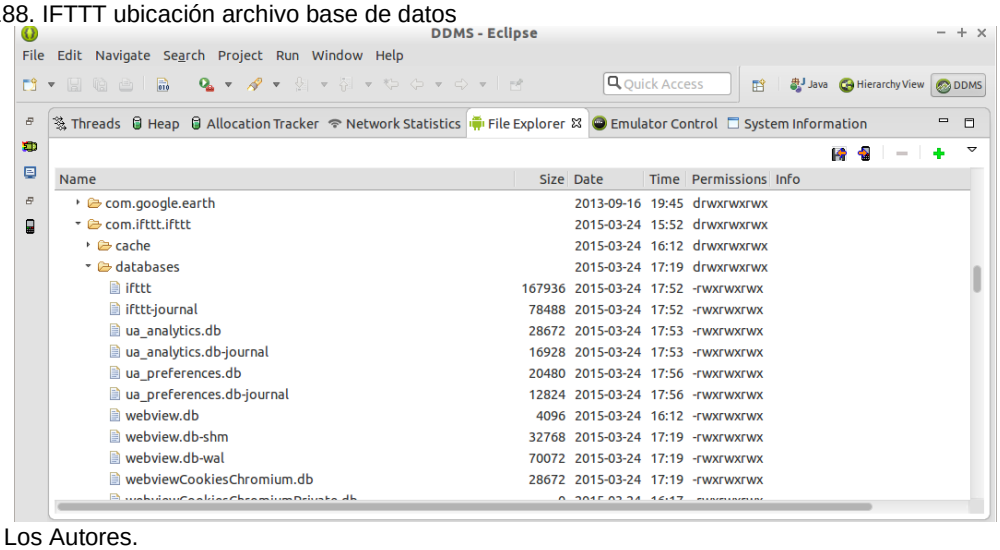
Figura 187. IFTTT ubicación shared_prefs



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “ifttt”, “ua-analytics.db”, “ua-analytics.db-journal”, “uapreferences.db”, “uapreferences.db-journal”, “uapreferences.db”, “uapreferences.db-journal”, “webviews.db” y “webviewsCookiesChromium.db” las cuales son usadas por las API y no contienen información sensible de la aplicación.

Figura 188. IFTTT ubicación archivo base de datos



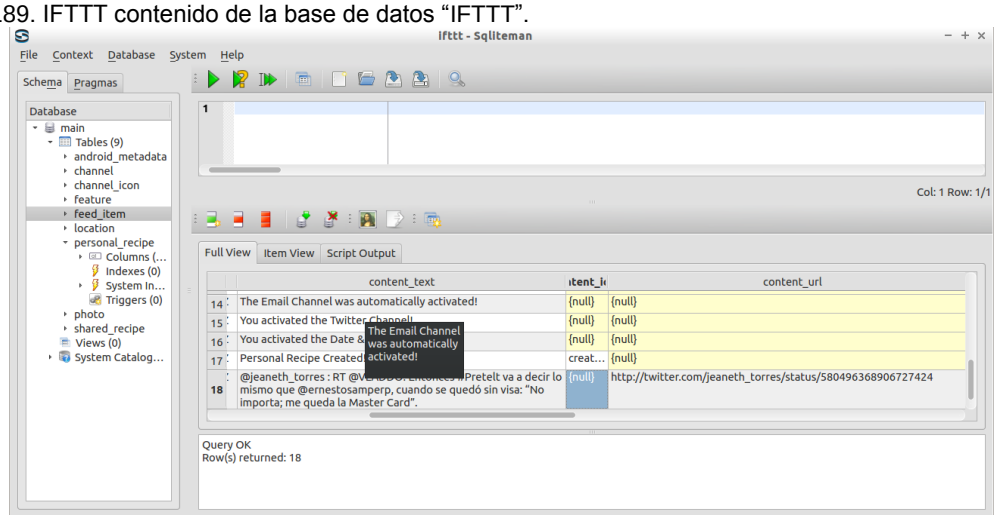
Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “iftttt” se muestra en la tabla *feed_item* que almacena los mensajes que cumplen con las recetas configuradas.

La información encontrada no se considera sensible por cuanto no se observa almacenamiento de datos del usuario.

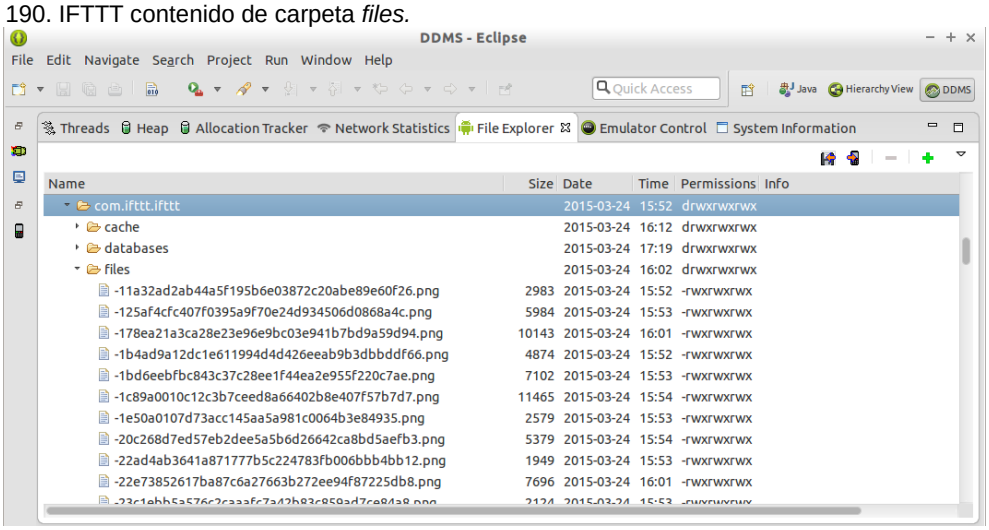
Figura 189. IFTTT contenido de la base de datos “IFTTT”.



Fuente: Los Autores.

En la carpeta files no se encuentran archivos que manejen información sensible.

Figura 190. IFTTT contenido de carpeta *files.*

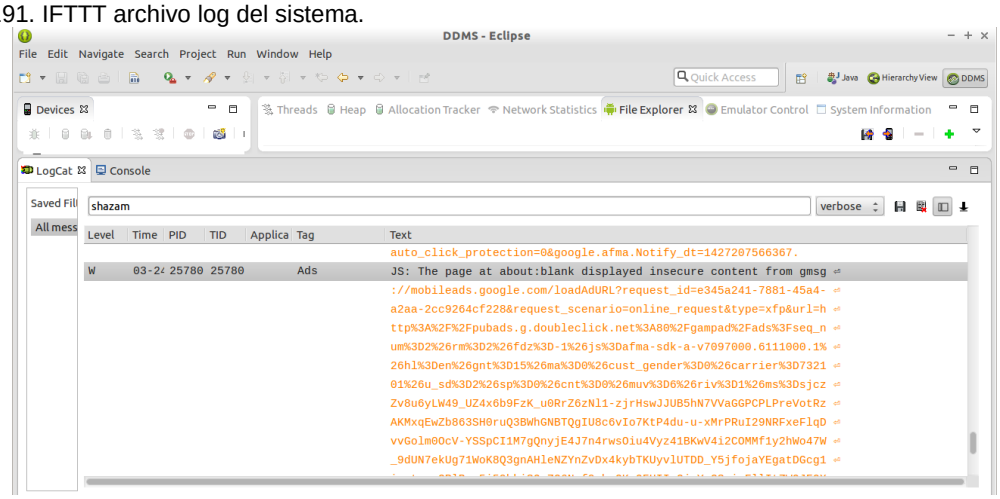


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log se observa el almacenamiento de información cifrada, pero no es posible ver su contenido

Figura 191. IFTTT archivo log del sistema.

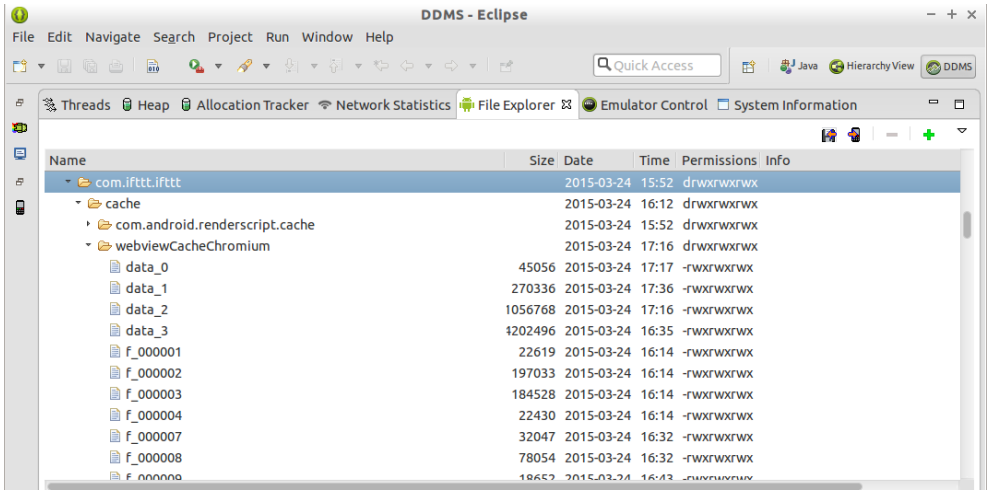


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación en donde no se encontró almacenamiento de información sensible.

Figura 192. IFTTT contenido de la cache.

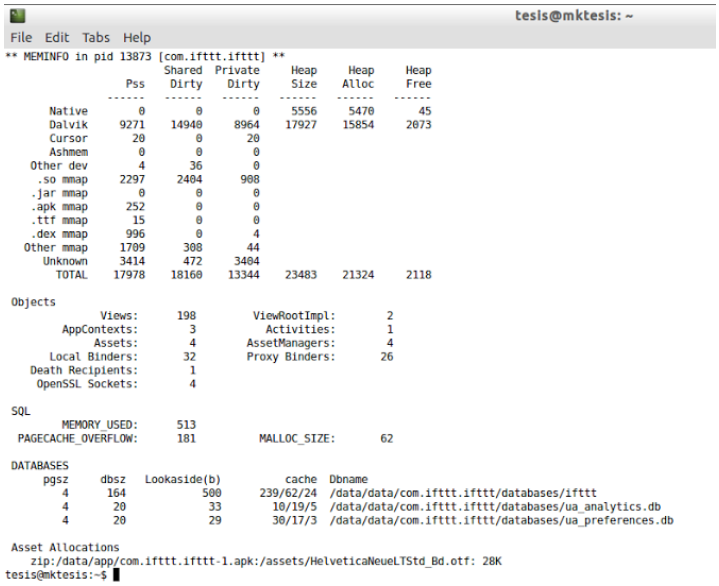


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información sensible de la misma.

Figura 193. IFTTT información de la memoria.



Fuente: Los Autores.

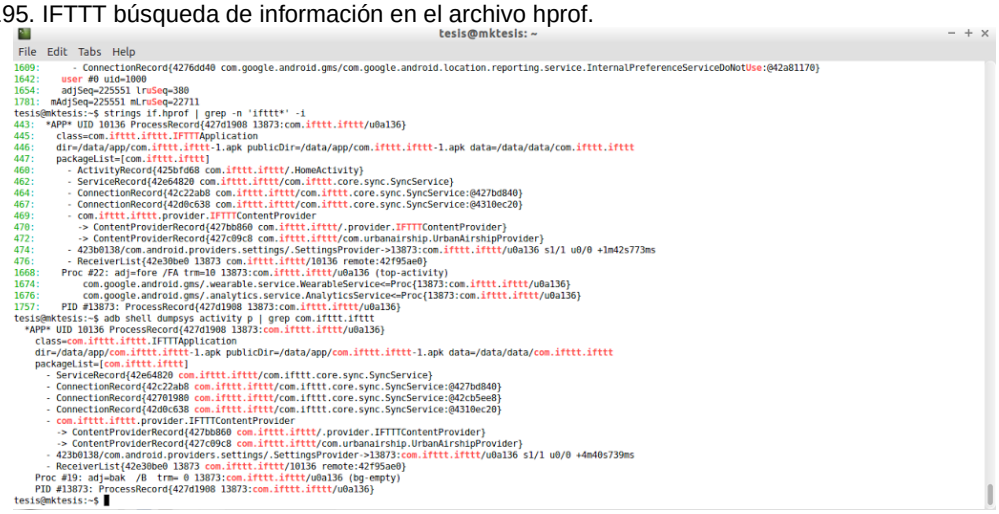
Figura 194. IFTTT búsqueda de información en la memoria.

Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin identificar información sensible.

Figura 195. IFTTT búsqueda de información en el archivo hprof.



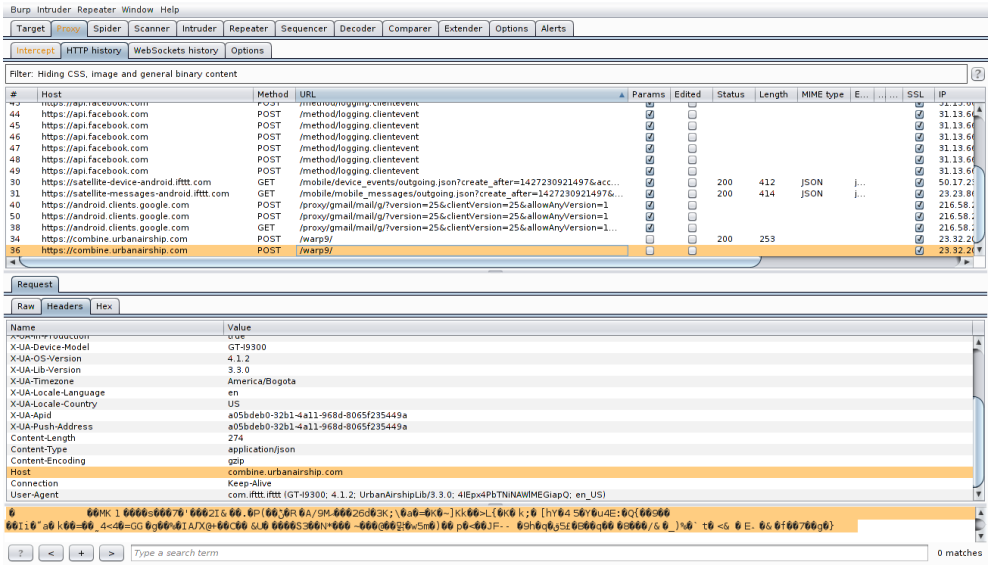
Fuente: Los Autores.

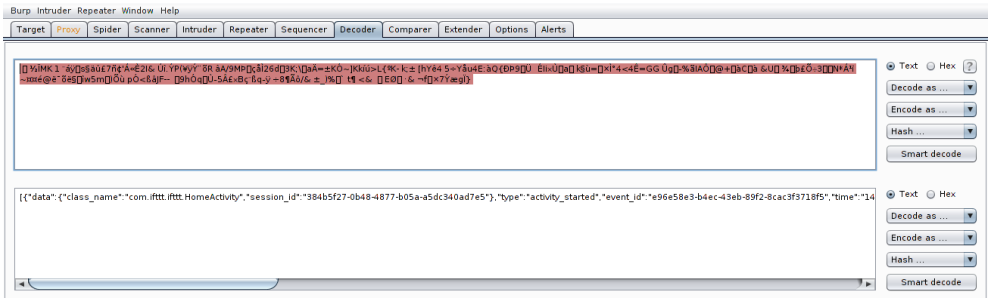
M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Realizada la interceptación del tráfico de la red se encontró que la aplicación utiliza protocolos HTTP y SSL en donde cifra la información transmitida cuando el login se realiza mediante el registro en la aplicación.

Figura 196. IFTTT información transmitida cifrada.

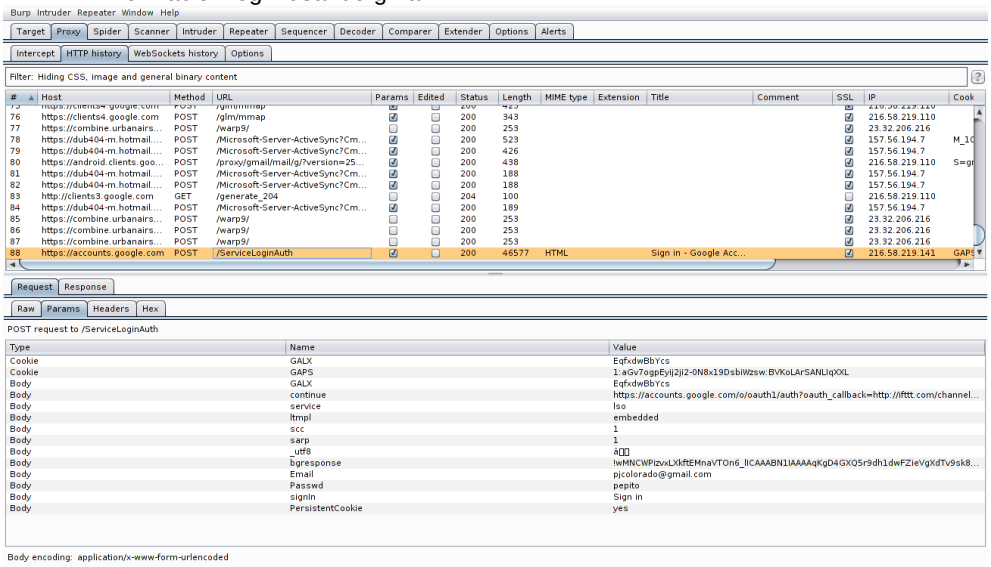




Fuente: Los Autores.

Se realiza login con la cuenta de correo electrónico de gmail identificándose la información sensible del usuario (correo electrónico) y la contraseña.

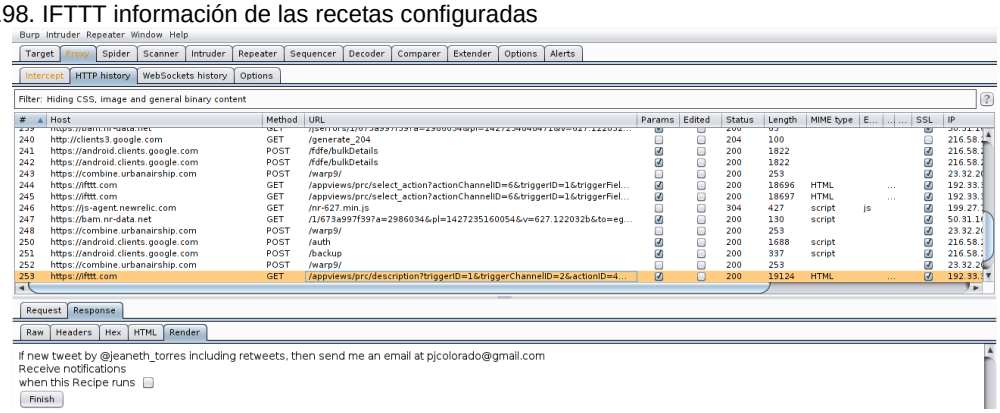
Figura 197. IFTTT información login usando gmail.



Fuente: Los Autores.

Se realiza la configuración de una receta para redes sociales identificándose la combinación dada.

Figura 198. IFTTT información de las recetas configuradas

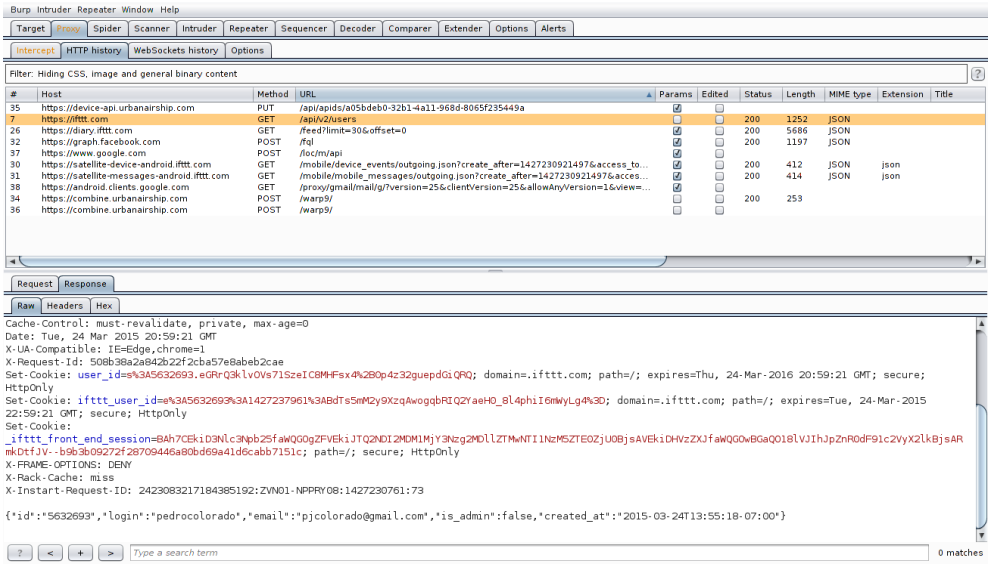


Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

Al iniciar la aplicación e ingresar los datos de login se observa la comunicación insegura por cuanto se muestran datos sensibles como el usuario y el correo electrónico.

Figura 199. IFTTT información del inicio de sesión



Fuente: Los Autores.

VIII. Aplicación Groupon

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
Groupon (com.groupon)

2- Funcionalidad básica
Aplicación para comprar productos o servicios que se encuentra con descuento, permitiendo obtener un ahorro significativo. Permite comprar, administrar y canjear los groupones, navegar y acceder a las ofertas del día de la ciudad de ubicación.

3- ¿La aplicación realiza transacciones electrónicas?

[X] Si [ ] No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[X] Si [ ] No

Figura 200. Groupon Permisos

Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

[ ] NFC
[X] GPS
[ ] Micrófono
[X] USB
[ ] Bluetooth
[X] Cámara
[X] Sensores

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

[ ] Telefonía (SMS, teléfono)
[ ] Recepción de datos de aplicaciones y otros
[ ] Contactos
[ ] Google Wallet

	servicios en el dispositivo		
X	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)		Correo electrónico
	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 201. Groupon interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[X] Si [ ] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[ ] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

B- Análisis estático

General

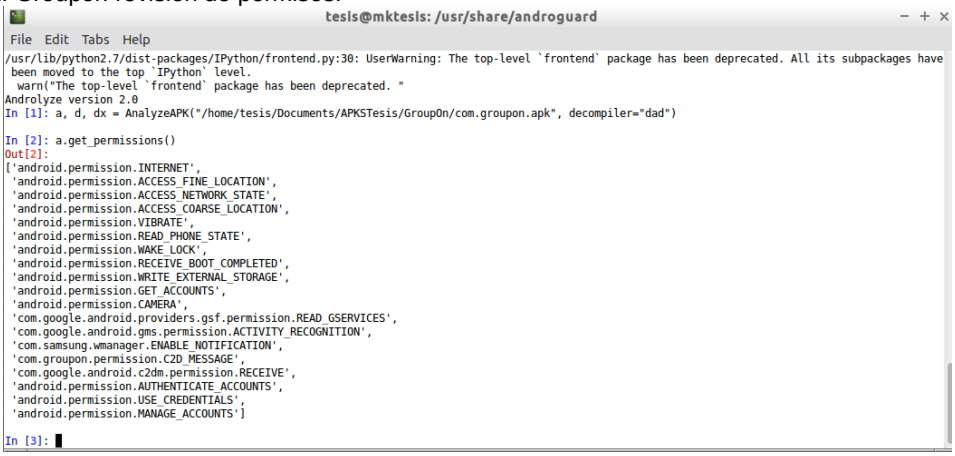
1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- ACCESS_COARSE_LOCATION permite que una aplicación acceda a la ubicación aproximada derivado de las fuentes de ubicación de red, como las redes telefonía móvil y Wi - Fi.

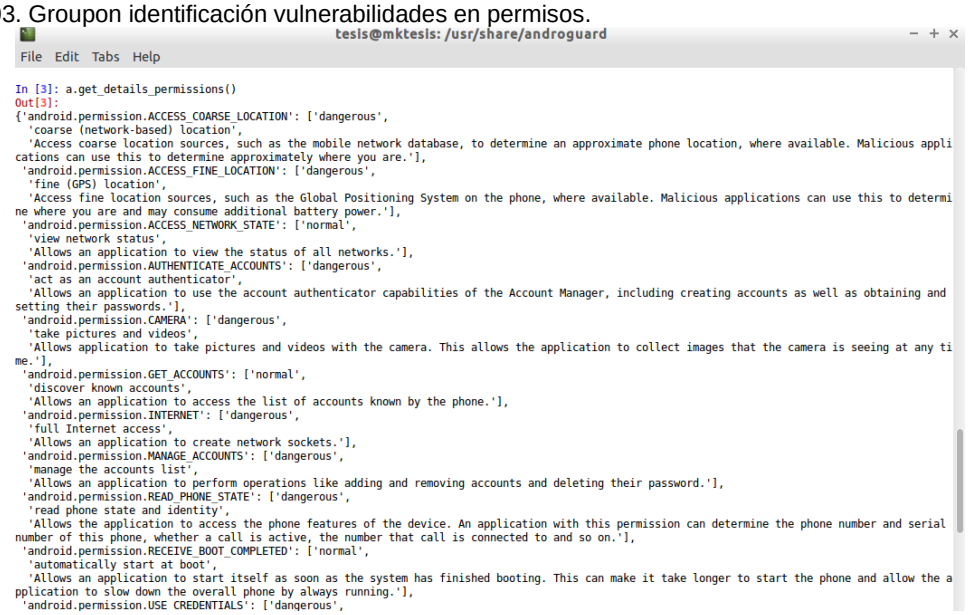
- ACCESS_FINE_LOCATION permite que una aplicación acceda a la ubicación precisa de las fuentes de ubicación, como el GPS, las redes de telefonía móvil y Wi - Fi.
- AUTHENTICATE_ACCOUNTS permite que una aplicación pueda actuar como administrador de cuentas para la autenticación
- CAMERA permite acceder a la cámara.
- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.
- MANAGE_ACCOUNTS permite que una aplicación pueda gestionar la lista de cuentas de administrador de cuentas.
- READ_PHONE_STATE permite acceso de sólo lectura al estado del teléfono.
- USE_CREDENTIALS permite que una aplicación solicite tokens de autenticación a partir del administrador de cuentas.

Figura 202. Groupon revisión de permisos.



Fuente: Los Autores.

Figura 203. Groupon identificación vulnerabilidades en permisos.



Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

No. Realizada la revisión del código fuente no se encontró uso de métodos de validación de este parámetro en la búsqueda de instrucciones con los comandos “xbin”, “su”, “sbin”, “system”.

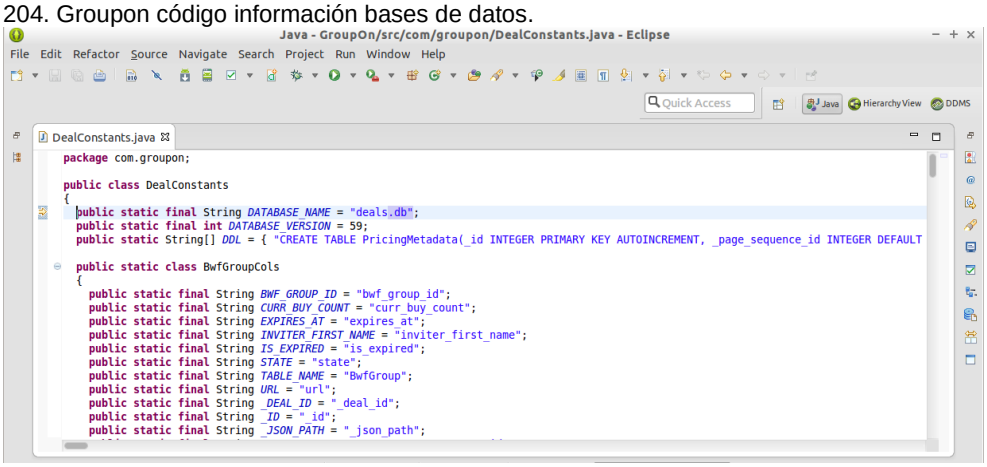
Los dispositivos rooteados no incluyen todas las protecciones de seguridad en el sistema operativo permitiendo el acceso total a información y datos de aplicaciones.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa una base de datos llamado *deal.db* y *deals_v2.db*, el archivo que muestra esta información es:
com/groupon/DealConstants.java
com/groupon/v2/db/GrouponOrmLiteHelperV2.java

Figura 204. Groupon código información bases de datos.



Fuente: Los Autores.

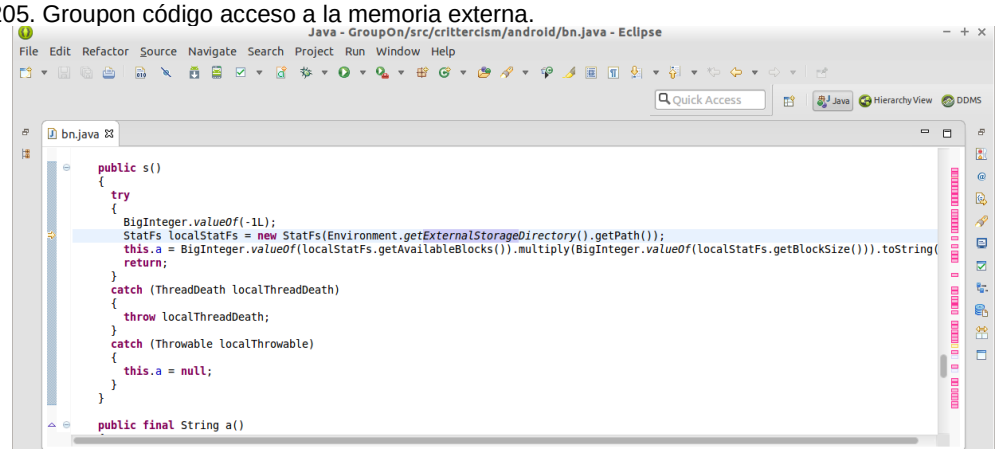
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 205. Groupon código acceso a la memoria externa.

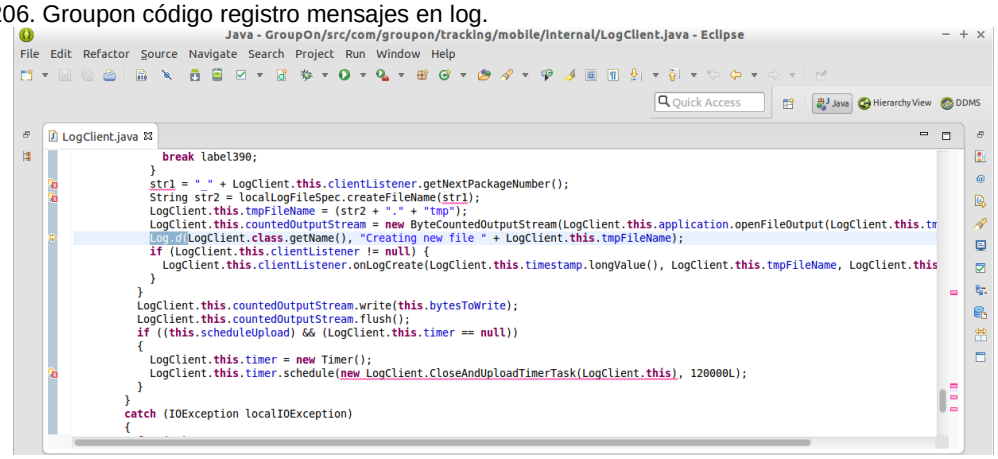


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 206. Groupon código registro mensajes en log.



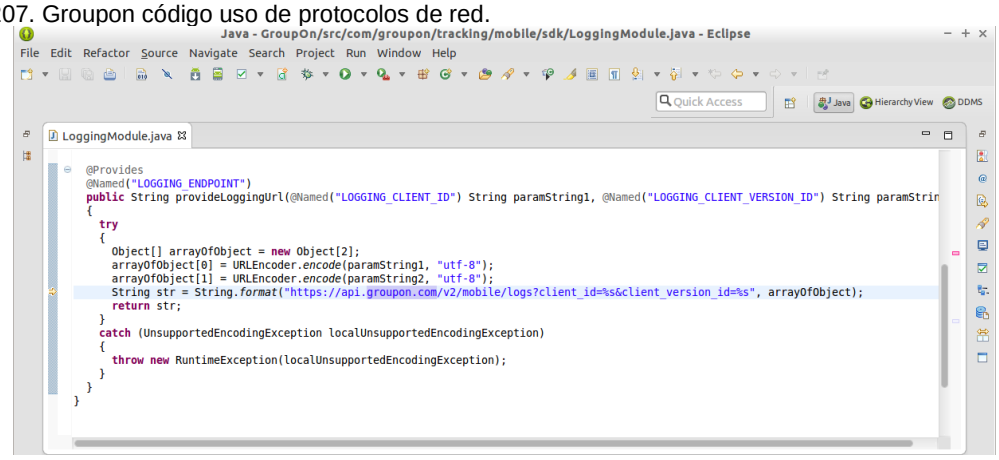
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 207. Groupon código uso de protocolos de red.

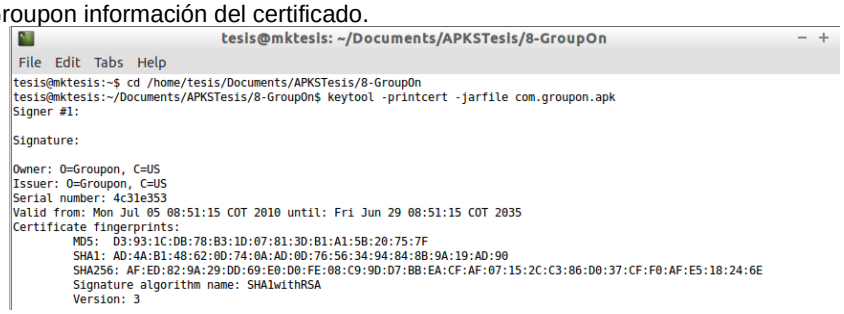


Fuente: Los Autores.

7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

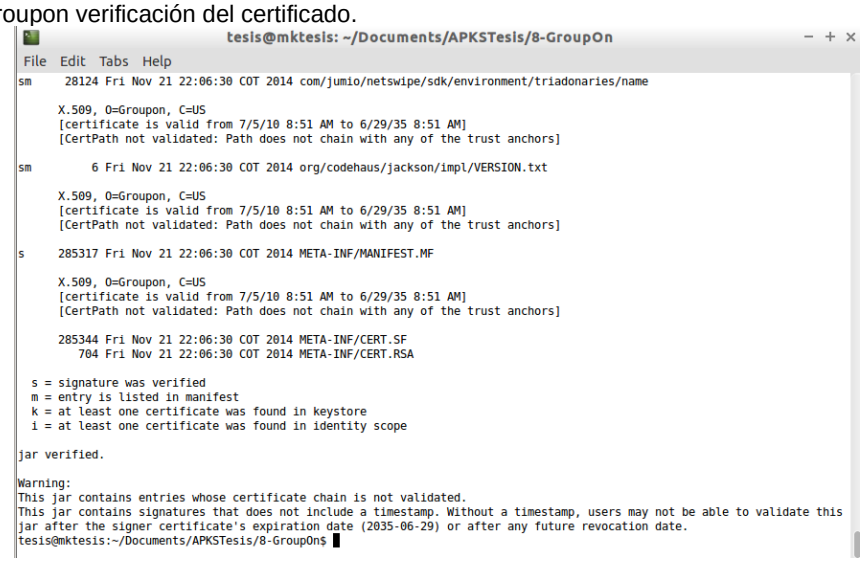
Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 208. Groupon información del certificado.



Fuente: Los Autores.

Figura 209. Groupon verificación del certificado.



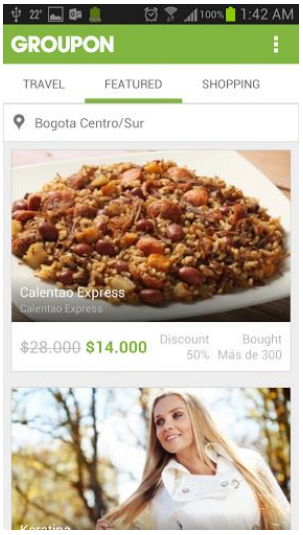
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 210. Groupon configuración preferencias



Fuente: Los Autores.

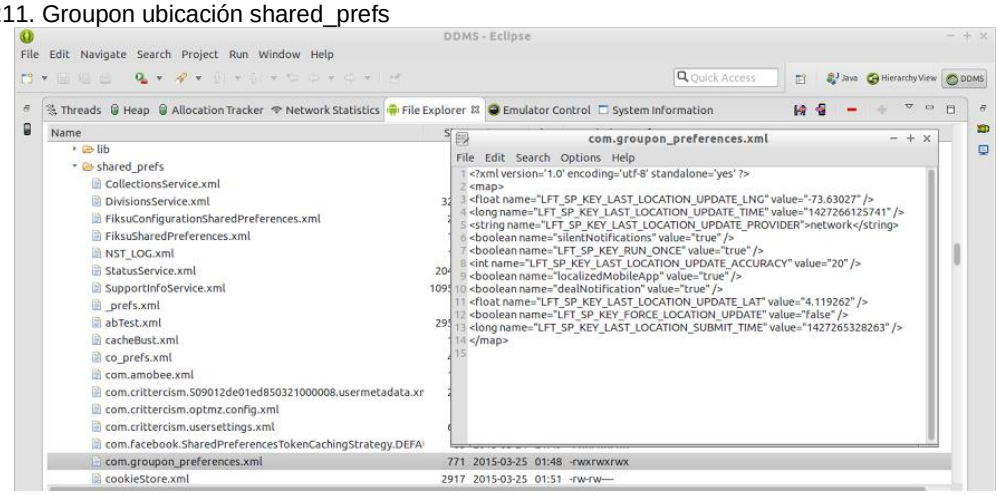
M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.groupon” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

Se observa en la carpeta “/shared_prefs”, donde se revisa los archivos “xml” encontrándose almacenamiento de información sensible como el correo electrónico del usuario con el cual inicia sesión en la aplicación.

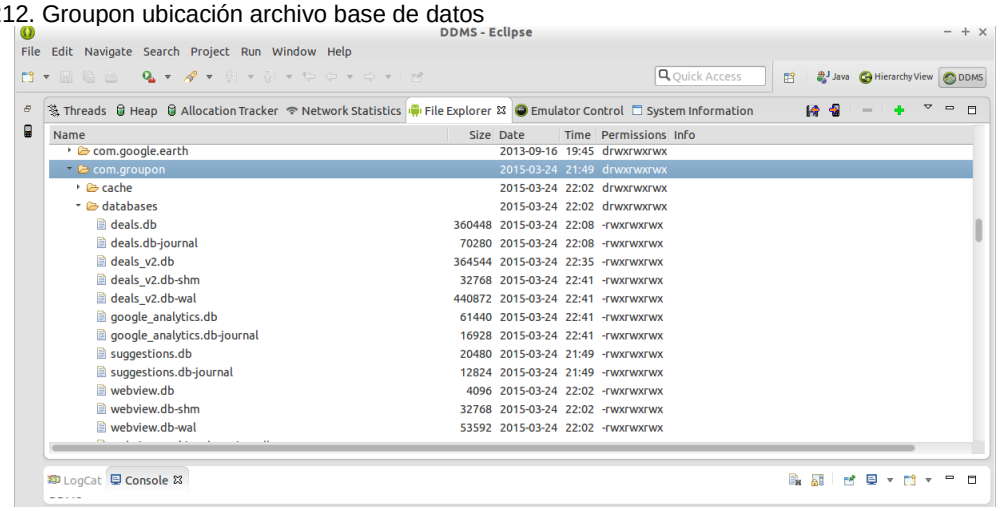
Figura 211. Groupon ubicación shared_prefs



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “deals.db”, “deals.db-journal”, “deals_v2.db”, “deals_v2.db-journal”, “google_analytics.db”, “google_analytics.db-journal”, “suggestions.db” y “webview.db”, las cuales son usadas por las API y no contienen información sensible de la aplicación.

Figura 212. Groupon ubicación archivo base de datos

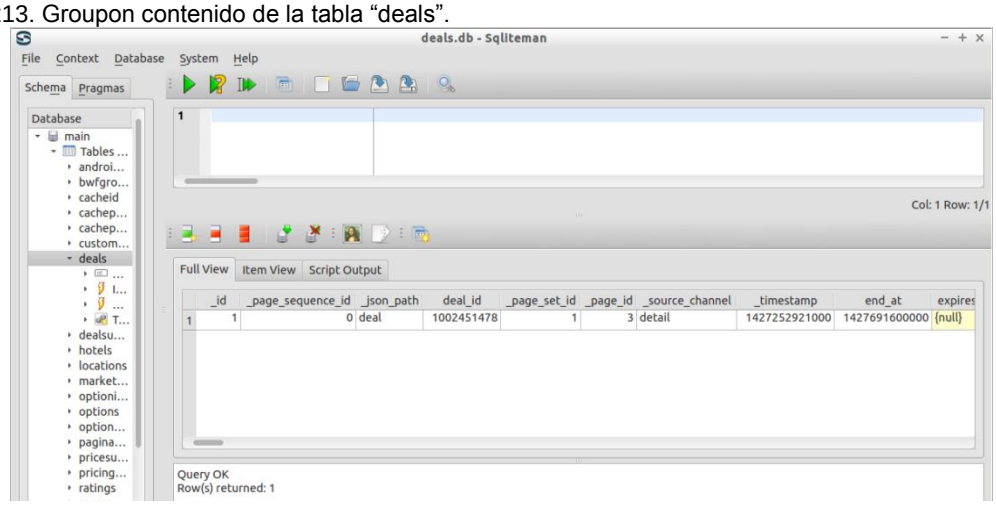


Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “deals.db” la información encontrada no se considera sensible por que no almacena información del usuario.

Figura 213. Groupon contenido de la tabla “deals”.



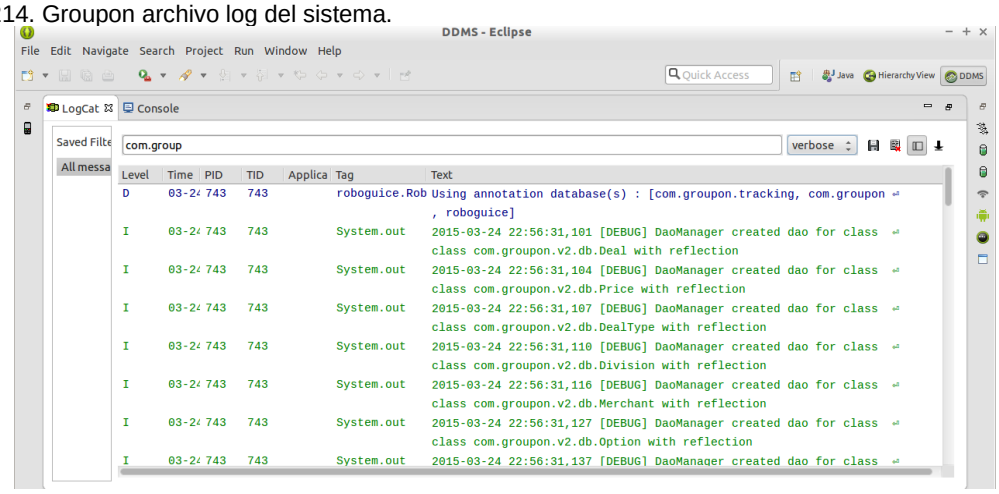
Fuente: Los Autores.

En la carpeta files no se encuentran archivos que manejen información sensible.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log no se observa el almacenamiento de información sensible.

Figura 214. Groupon archivo log del sistema.

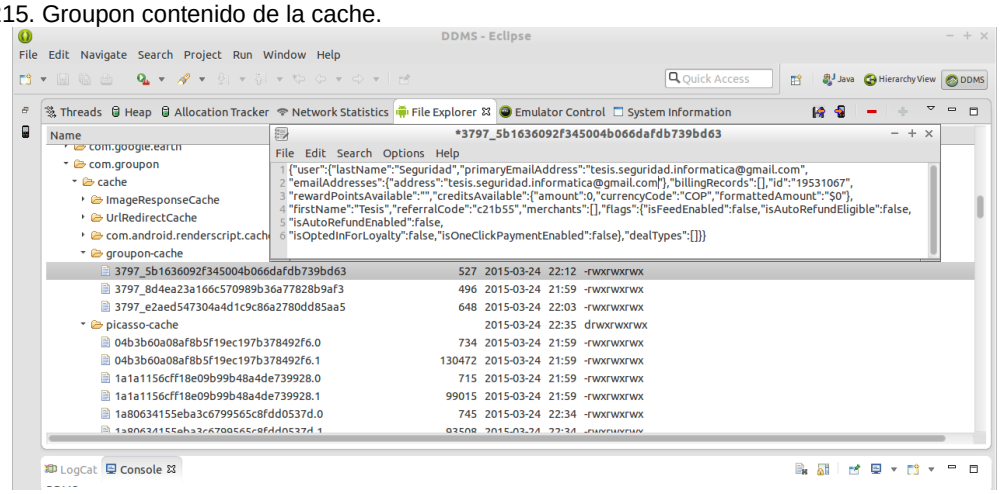


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación en donde se encontró almacenamiento de información sensible como es el correo electrónico, el nombre del usuario.

Figura 215. Groupon contenido de la cache.

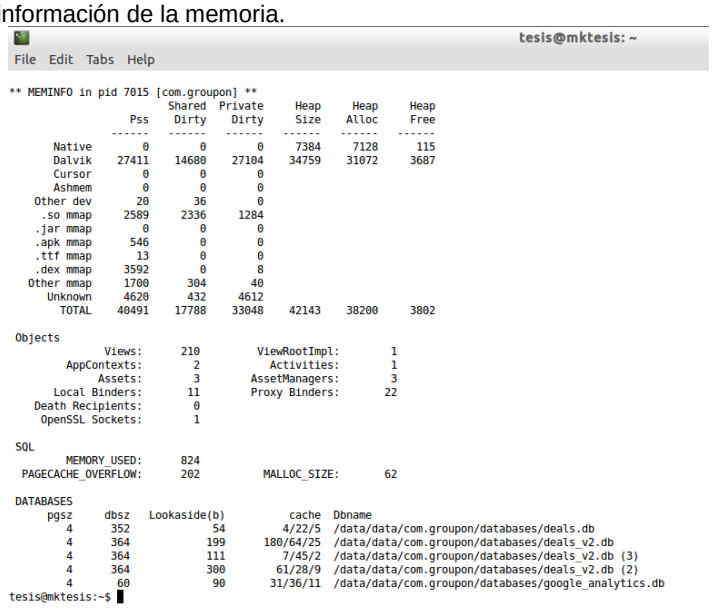


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información sensible de la misma.

Figura 216. Groupon información de la memoria.



Fuente: Los Autores.

Figura 217. Groupon búsqueda de información en la memoria.

Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin identificar información sensible.

Figura 218. Groupon búsqueda de información en el archivo hprof.

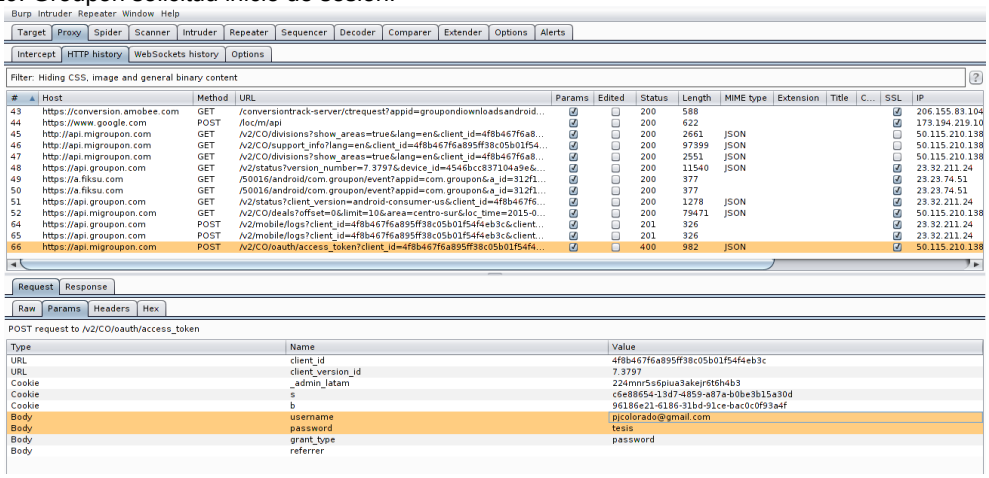
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Se realiza el loggeo en la aplicación logrando interceptarse el correo electrónico y la contraseña con el cual el usuario inicia sesión identificándose la transmisión de información sensible a través del protocolo HTTP y SSL.

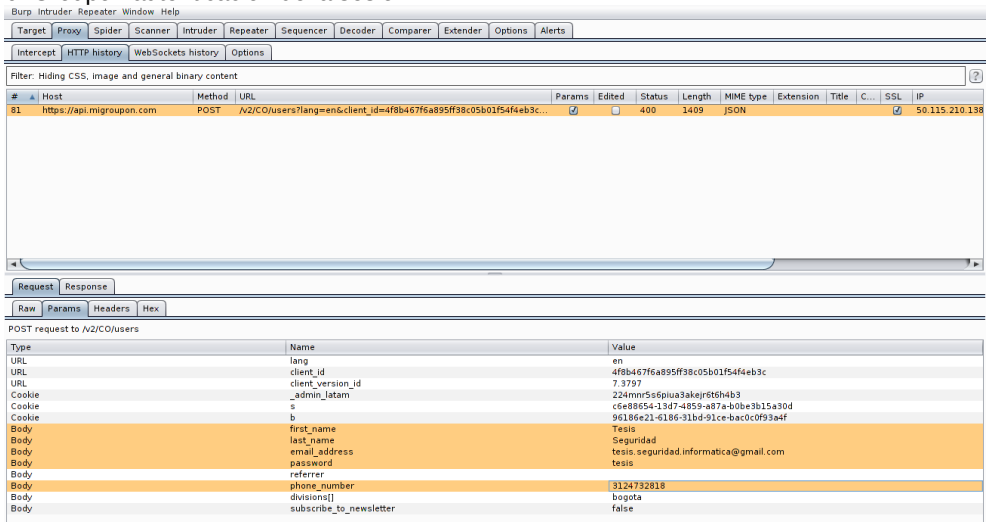
Figura 219. Groupon solicitud inicio de sesión.



Fuente: Los Autores.

Una vez completada la petición, se identifica el nombre de usuario, teléfono, correo electrónico y contraseña de acceso.

Figura 220. Groupon autenticación de la sesión.

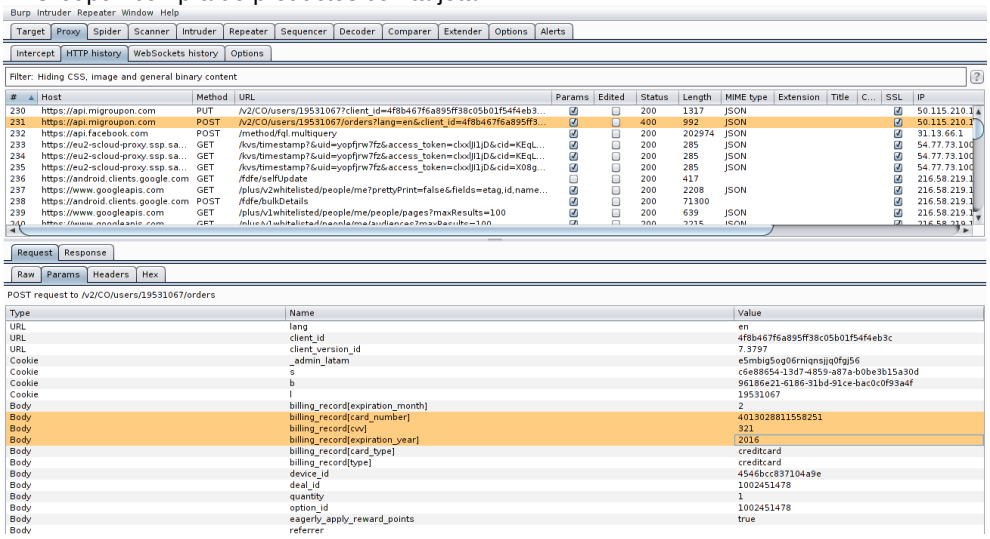


Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

Se realiza el proceso de compra de un producto ingresando la información de la tarjeta de crédito, logrando interceptarse la información que es enviada sin cifrar aun cuando usan protocolos de comunicación segura HTTPS y SSL.

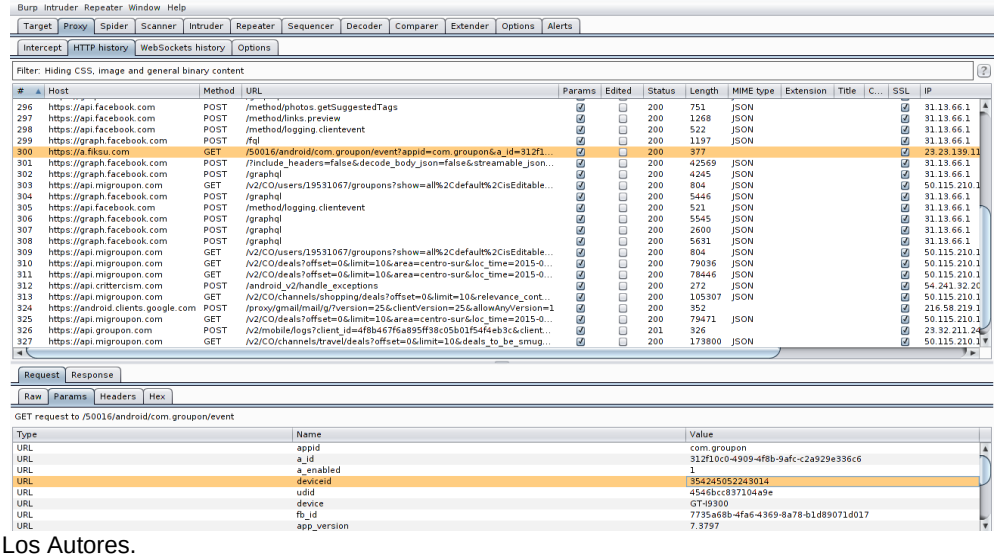
Figura 221. Groupon compra de productos con tarjeta



Fuente: Los Autores.

Adicionalmente se observa que en la transmisión de información cuando se loggea o comparte información se envía el IMEI del dispositivo móvil.

Figura 222. Groupon envío IMEI del teléfono.



Fuente: Los Autores.

IX. Aplicación Locket Lock Screen for English

A continuación se describen los resultados de la evaluación de la aplicación.

A- Recopilación de información sobre la Aplicación

1- Nombre
Locket Lock Screen for English (com.locket.matterhorn)

2- Funcionalidad básica
Aaplicación de pantalla de bloqueo inteligente que ayuda a aprender inglés a través de historias de tendencias que se muestran cada vez que se desbloquea el dispositivo móvil.

3- ¿La aplicación realiza transacciones electrónicas?

[ ]	Si	[X]	No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

[ ]	Si	[ ]	No

Figura 223. Locket Permisos

Fuente: Los Autores.

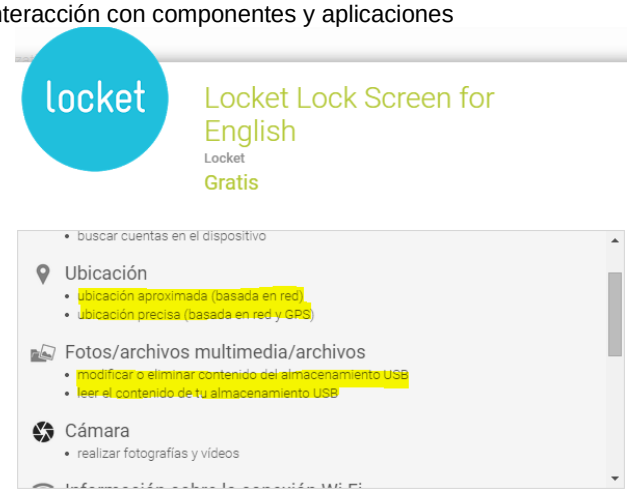
4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

[ ]	NFC	[ ]	Bluetooth
[X]	GPS	[X]	Cámara
[ ]	Micrófono	[X]	Sensores
[X]	USB		

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

☐	Telefonía (SMS, teléfono)	☐	Contactos
☐	Recepción de datos de aplicaciones y otros servicios en el dispositivo	☐	Google Wallet
☐	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)	☐	Correo electrónico
☐	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 224. Locket interacción con componentes y aplicaciones



Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

☒ Si ☐ No

7- Identificar las interfaces de red inalámbrica utilizadas:

☒ Wi-Fi (802.11) ☐ NFC ☐ Bluetooth

B- Análisis estático

General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

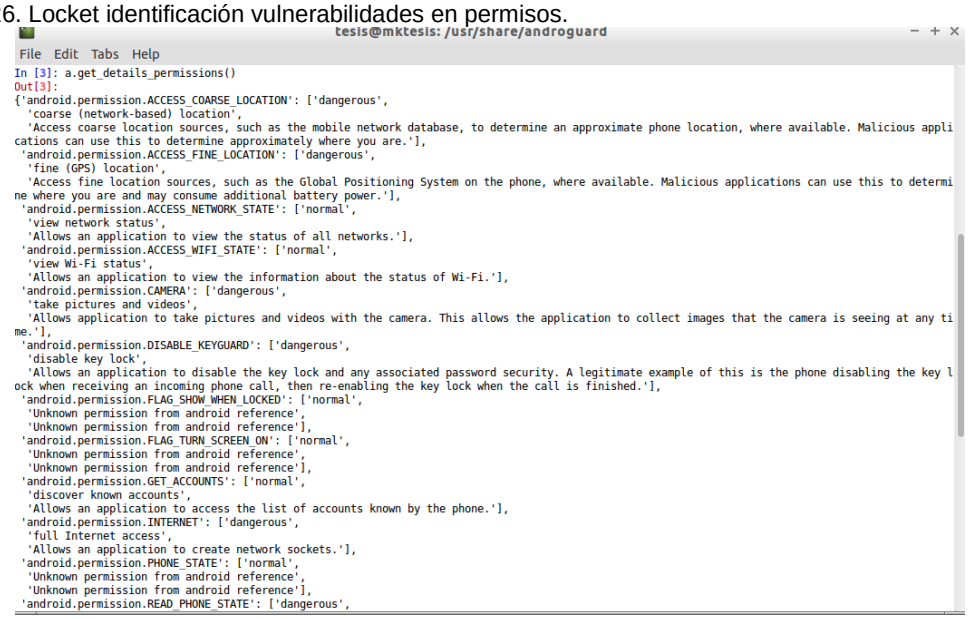
El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- ACCESS_COARSE_LOCATION permite que una aplicación acceda a la ubicación aproximada derivado de las fuentes de ubicación de red, como las redes telefonía móvil y Wi - Fi.
- ACCESS_FINE_LOCATION permite que una aplicación acceda a la ubicación precisa de las fuentes de ubicación, como el GPS, las redes de telefonía móvil y Wi - Fi.
- CAMERA permite acceder a la cámara.
- DISABLE_KEYGUARD permite a las aplicaciones desactivar el bloqueo del teclado.
- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.
- READ_PHONE_STATE permite acceso de sólo lectura al estado del teléfono.

Figura 225. Locket revisión de permisos.

Fuente: Los Autores.

Figura 226. Locket identificación vulnerabilidades en permisos.



Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

No. Realizada la revisión del código fuente no se encontró uso de métodos de validación de este parámetro en la búsqueda de instrucciones con los comandos “xbin”, “su”, “sbin”, “system”.

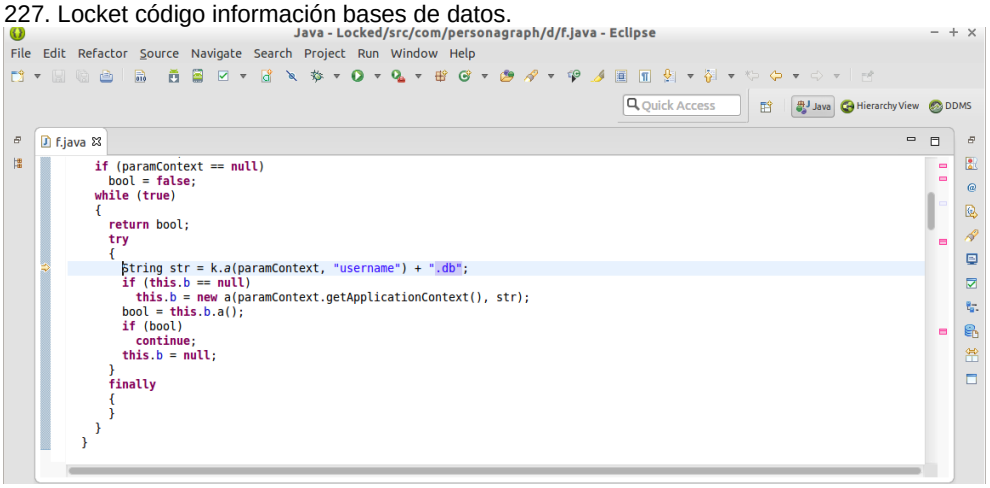
Los dispositivos rooteados no incluyen todas las protecciones de seguridad en el sistema operativo permitiendo el acceso total a información y datos de aplicaciones.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa base de datos dinámicas que se componen del nombre de usuario y la extensión “*.db”* y una base de datos denominada “*gtm_uris.db*” el archivo que muestra esta información es:
com/personagraph/d/f.java
com/google/tagmanager/PersistentHistore.java

Figura 227. Locket código información bases de datos.



Fuente: Los Autores.

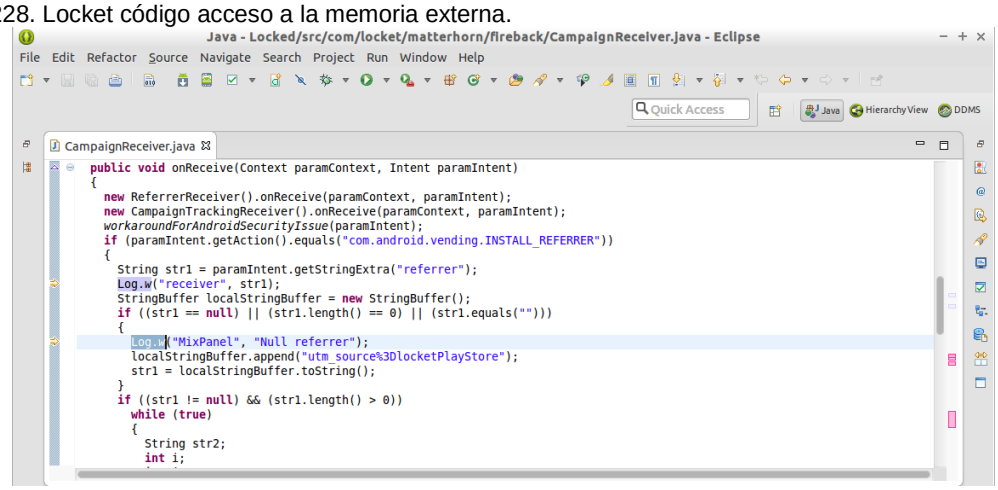
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 228. Locket código acceso a la memoria externa.

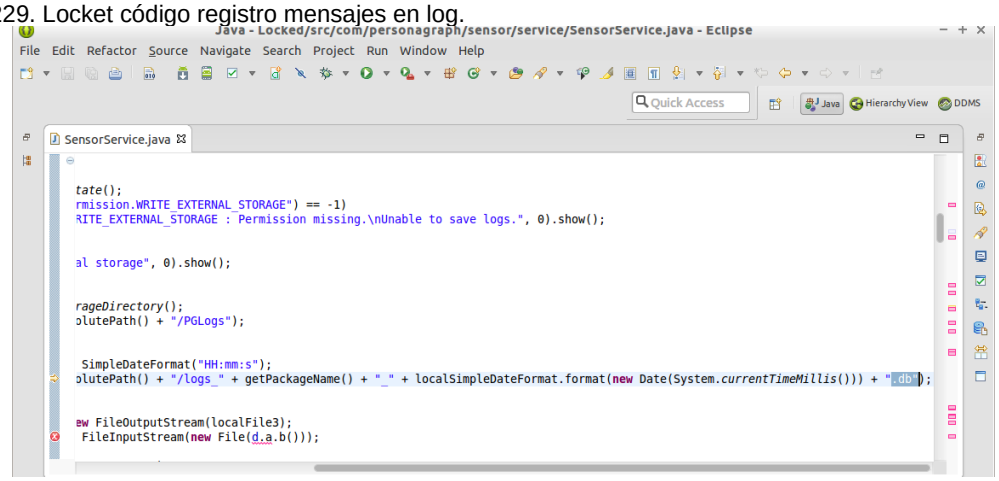


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 229. Locket código registro mensajes en log.



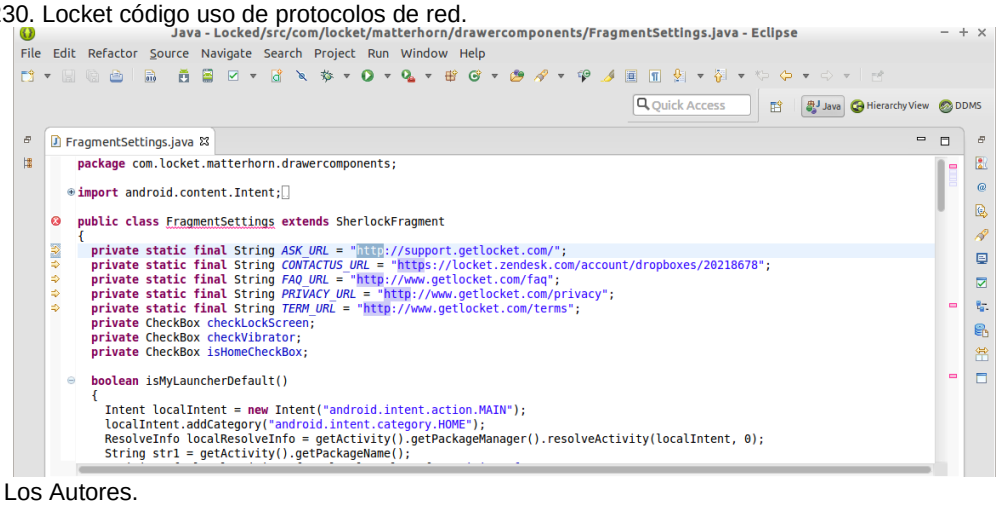
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 230. Locket código uso de protocolos de red.

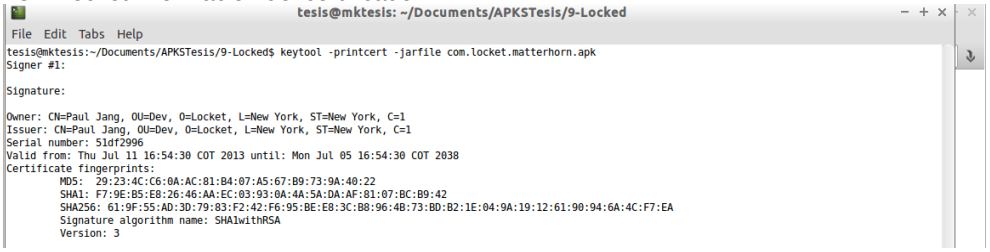


Fuente: Los Autores.

7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 231. Locket información del certificado.



Fuente: Los Autores.

Figura 232. Locket verificación del certificado.

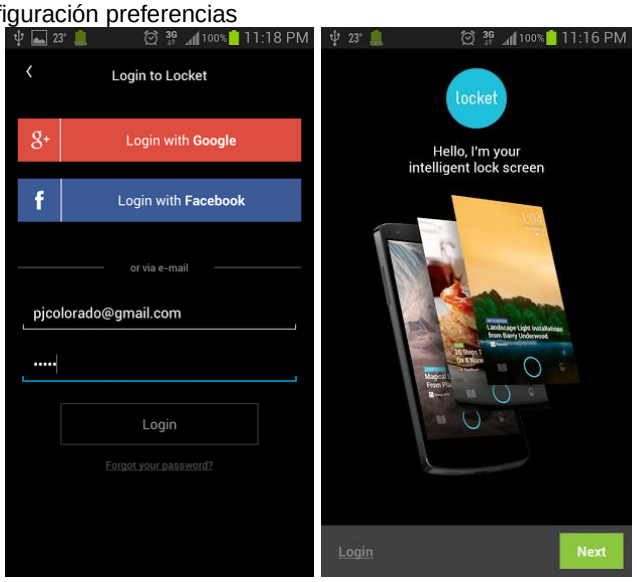
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 233. Locket configuración preferencias



Fuente: Los Autores.

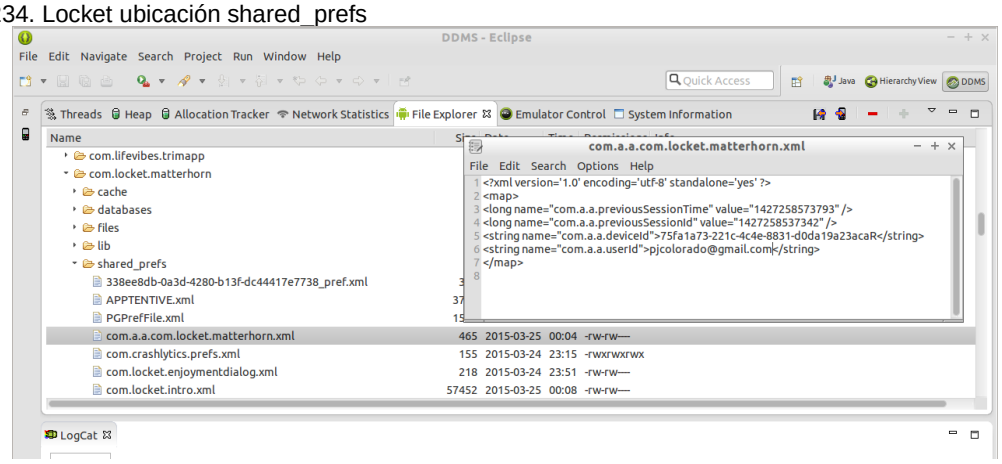
M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.Locket” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

Se observa en la carpeta “/shared_prefs”, donde se revisa los archivos “xml” encontrándose almacenamiento de información sensible como el correo electrónico del usuario con el cual inicia sesión en la aplicación.

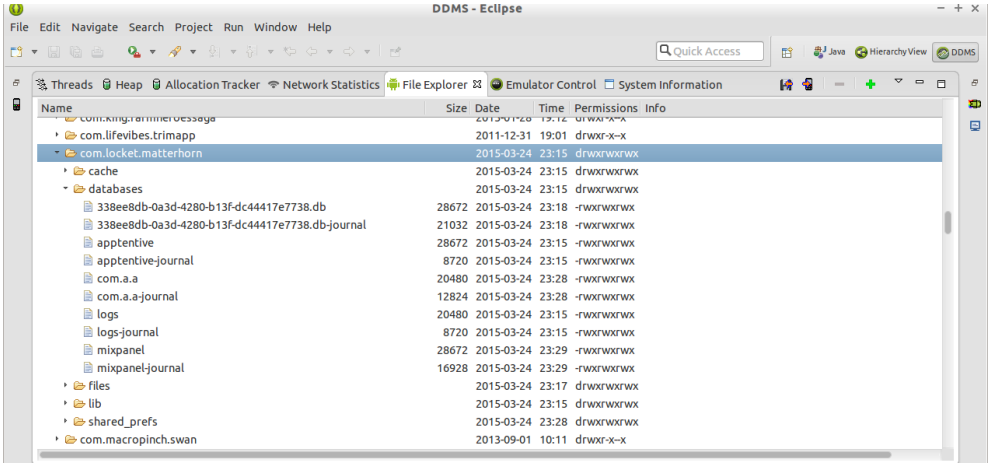
Figura 234. Locket ubicación shared_prefs



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “apptentive”, “apptentive-journal”, “com.a.a”, “com.a.a-journal”, “ “logs-journal”, “mixpanel” y “mix-panel-journal”, las cuales no contienen información sensible de la aplicación.

Figura 235. Locket ubicación archivo base de datos

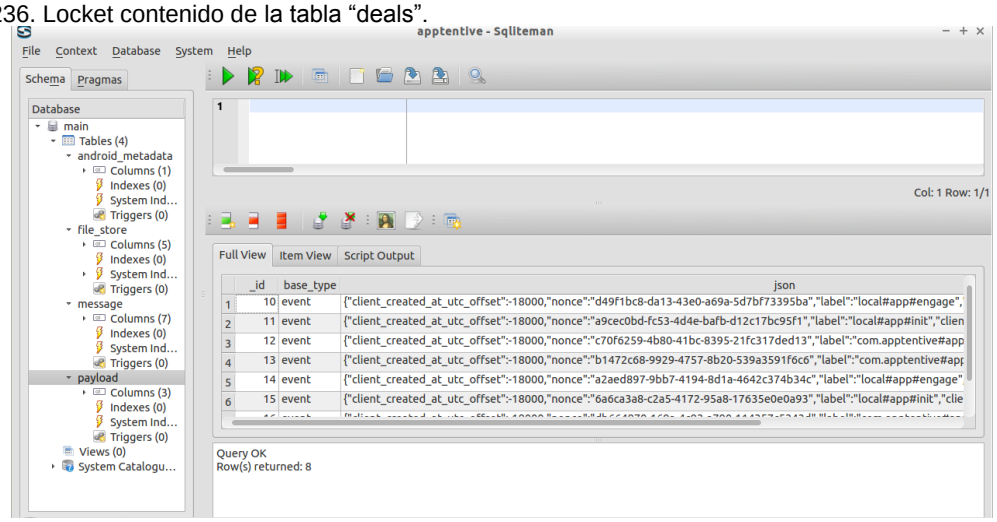


Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “apptentive” la información encontrada no se considera sensible por cuanto no se observa almacenamiento de datos se encuentra cifrado.

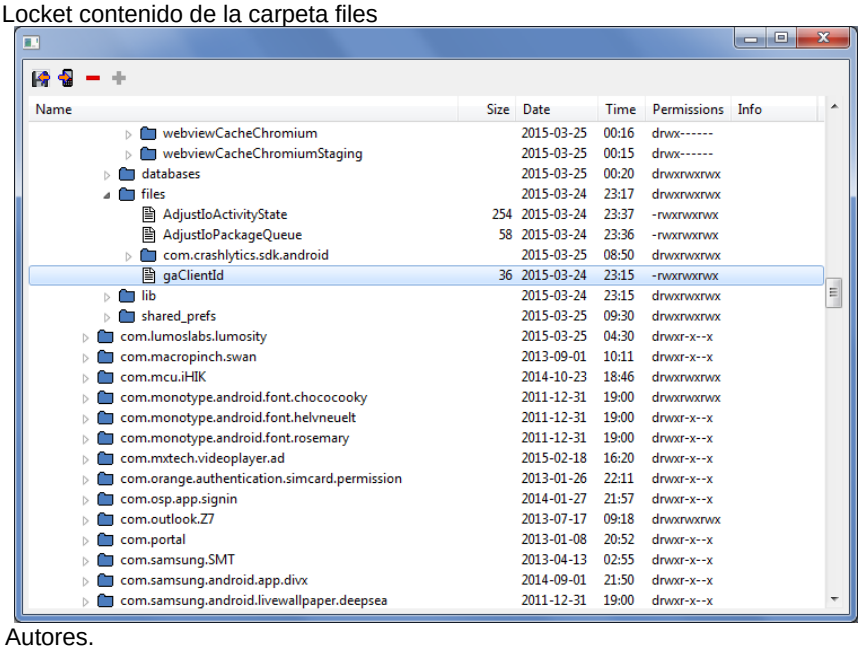
Figura 236. Locket contenido de la tabla “deals”.



Fuente: Los Autores.

En la carpeta files no se encuentran archivos que manejen información sensible.

Figura 237. Locket contenido de la carpeta files

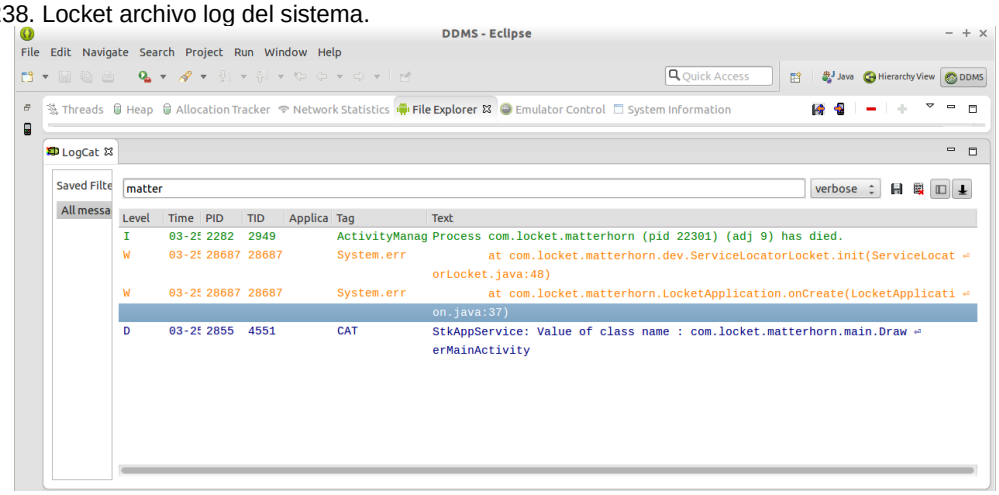


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log no se observa el almacenamiento de información sensible.

Figura 238. Locket archivo log del sistema.

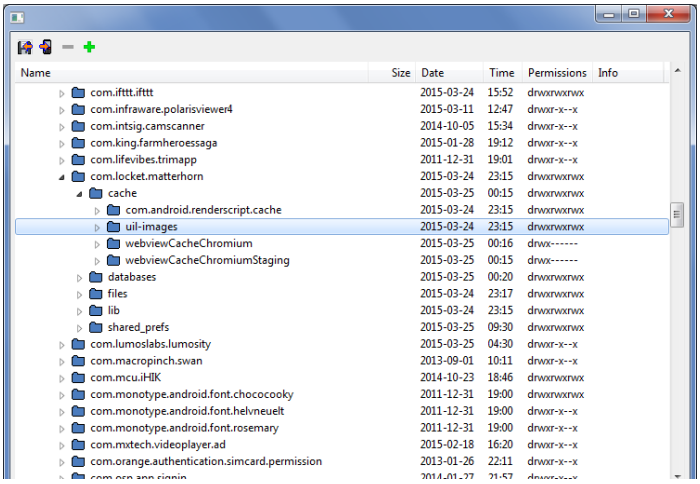


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación en donde no se encontró almacenamiento de información sensible.

Figura 239. Locket contenido de la cache.

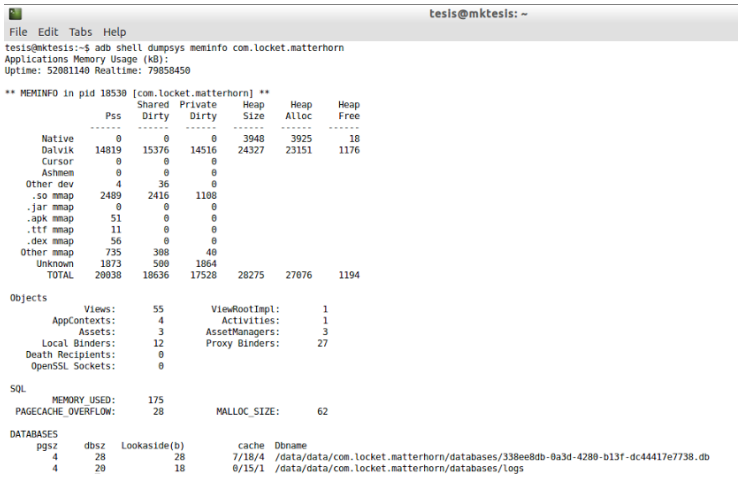


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

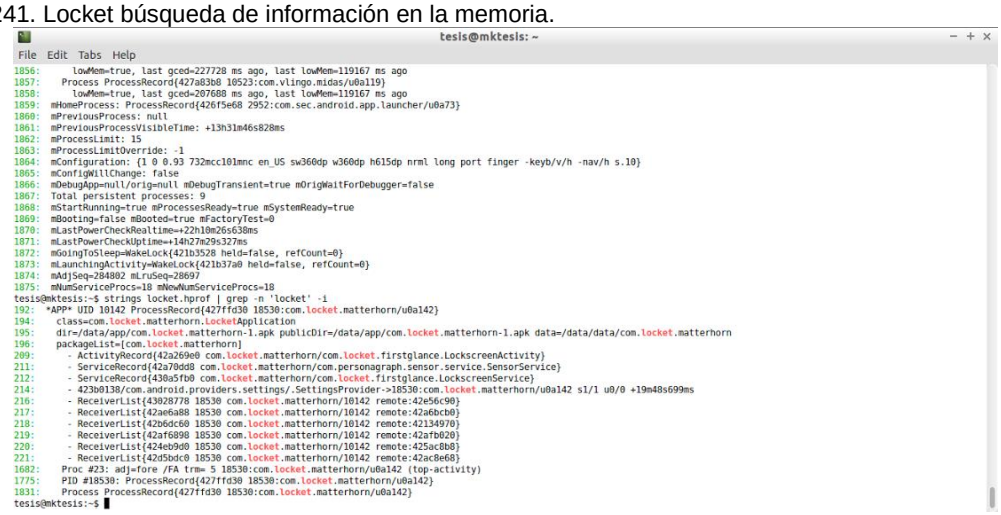
Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información sensible de la misma.

Figura 240. Locket información de la memoria.



Fuente: Los Autores.

Figura 241. Locket búsqueda de información en la memoria.



Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin identificar información sensible.

Figura 242. Locket búsqueda de información en el archivo hprof.

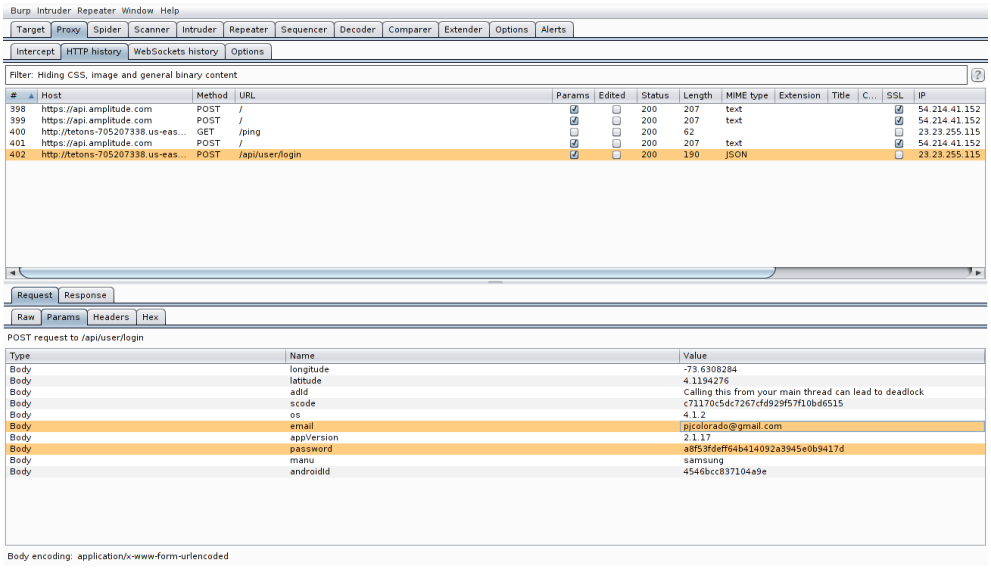
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Se realiza el loggeo en la aplicación logrando interceptarse el correo electrónico con el cual el usuario inicia sesión identificándose la transmisión de información sensible a través del protocolo HTTP.

Figura 243. Locket solicitud inicio de sesión.

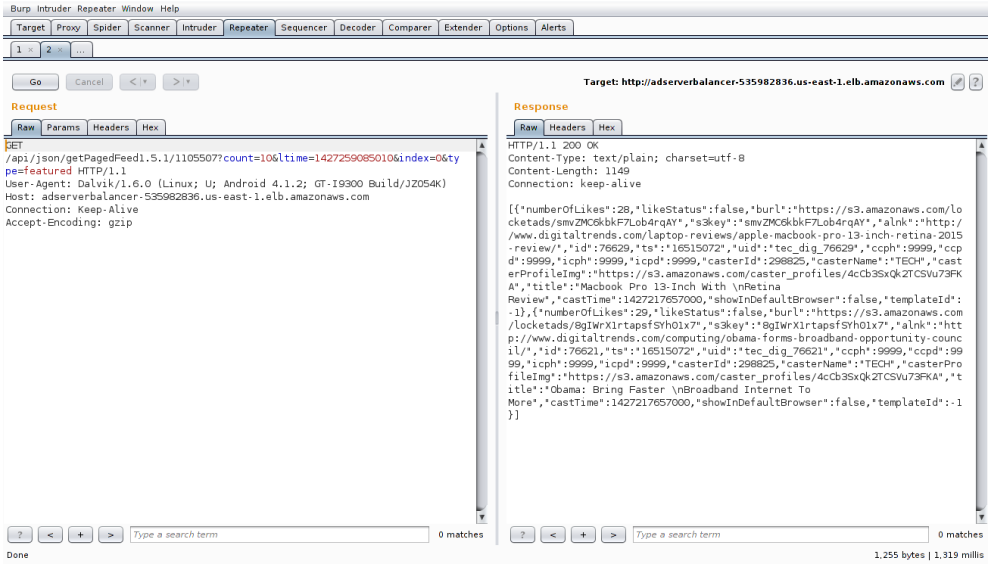


Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

No, se realiza el proceso de establecimiento de conexión una vez cerrada la cesión en la aplicación logrando obtener respuesta del servidor.

Figura 244. Locket establecimiento de conexión



Fuente: Los Autores.

X. Aplicación TimeHop

A- Recopilación de información sobre la Aplicación

1- Nombre
TimeHop (com.TimeHop)

2- Funcionalidad básica
Permite recuperar estados, fotos y mensajes de Facebook, Instagram, Twitter y Foursquare y reproduce en el dispositivo móvil pasado un día a la vez.r equiere login de Facebook y una vez en la aplicación puede vincularse con las cuentas de Twitter, Instagram y Foursquare.

3- ¿La aplicación realiza transacciones electrónicas?

☐ Si ☒ No

3.1 ¿Dentro de la aplicación se compran bienes o servicios?

☐ Si ☐ No

Figura 245. Timehop Permisos

Fuente: Los Autores.

4- La aplicación interactúa con alguno de los siguientes componentes de hardware:

☐ NFC ☐ Bluetooth

[ ]	GPS	[X]	Cámara
[ ]	Micrófono	[X]	Sensores
[X]	USB		

5- La aplicación interactúa con otras aplicaciones, servicios o datos como:

[X]	Telefonía (SMS, teléfono)	[ ]	Contactos
[ ]	Recepción de datos de aplicaciones y otros servicios en el dispositivo	[ ]	Google Wallet
[ ]	Redes sociales (Facebook, Twitter, LinkedIn, Google+, etc)	[ ]	Correo electrónico
[ ]	Almacenamiento en la nube (Google Drive, Dropbox, iCloud)		

Figura 246. Timehop interacción con componentes y aplicaciones

Fuente: Los Autores.

6- ¿La aplicación requiere registrar y/o configurar una cuenta de usuario destinada para las pruebas de auditoría?

[X] Si [ ] No

7- Identificar las interfaces de red inalámbrica utilizadas:

[ ] Wi-Fi (802.11) [ ] NFC [ ] Bluetooth

B- Análisis estático

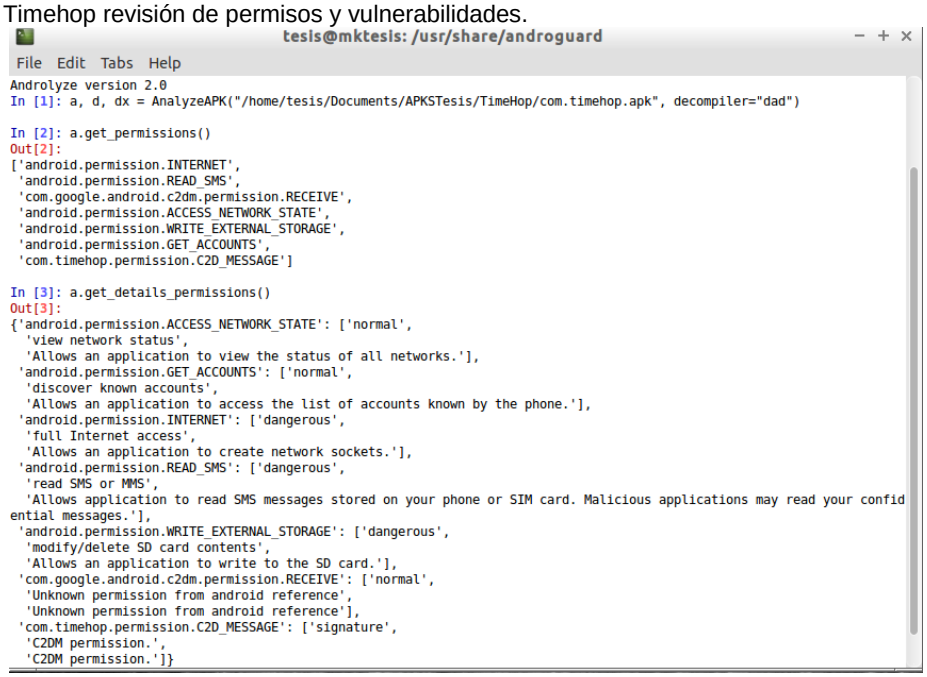
General

1- Revisar los permisos que la aplicación solicita en el archivo AndroidManifest.xml, así como los recursos autorizados.

El análisis de los permisos demuestra que algunos de ellos son de tipo “dangerous” lo cual representa un riesgo de seguridad.

- INTERNET permite establecer conexiones a través de internet, permitiendo el acceso total a través de la aplicación.
- READ_SMS permite leer los mensajes SMS.
- WRITE_EXTERNAL_STORAGE permite escribir información de la aplicación en medios externos permitiendo el acceso a los datos por cualquier otra aplicación.

Figura 247. Timehop revisión de permisos y vulnerabilidades.



Fuente: Los Autores.

2- ¿La aplicación valida si el dispositivo esta rooteado?

No. Realizada la revisión del código fuente no se encontró uso de métodos de validación de este parámetro en la búsqueda de instrucciones con los comandos “xbin”, “su”, “sbin”, “system”.

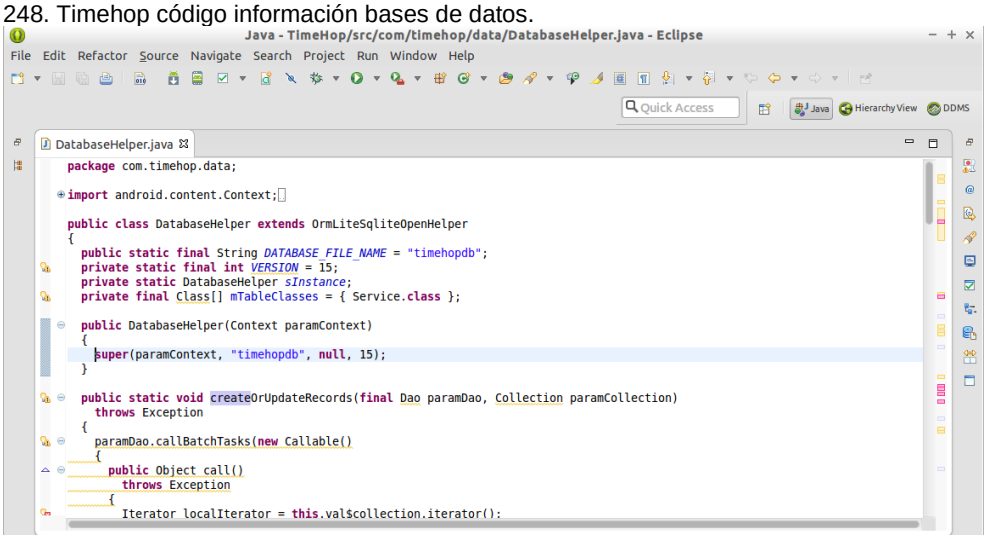
Los dispositivos rooteados no incluyen todas las protecciones de seguridad en el sistema operativo permitiendo el acceso total a información y datos de aplicaciones.

M2 Almacenamiento de datos inseguro

3- Determinar qué archivos y/o bases de datos utiliza la aplicación.

La revisión del código fuente del paquete muestra que la aplicación usa una base de datos llamado *timehopdb*, el archivo que muestra esta información es: *com/timeHop/data/DatabaseHelper.java*

Figura 248. Timehop código información bases de datos.



Fuente: Los Autores.

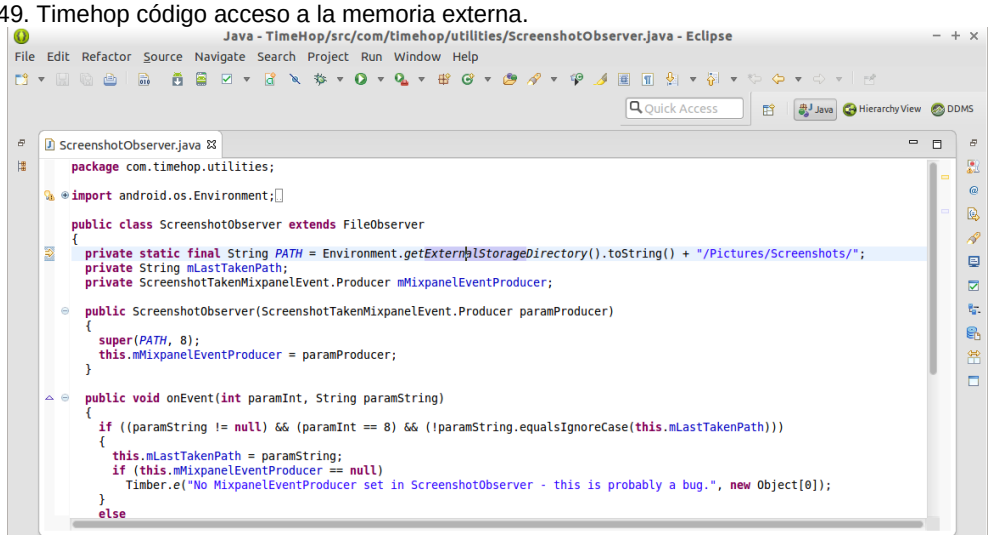
4- Identificar si la aplicación utiliza áreas de almacenamiento, fuera del SandBox, para guardar datos no encriptados como:

a) Ubicaciones con acceso limitado (SD card, directorios temporales, etc.).
b) Directorios que pueden terminar en copias de seguridad u otros lugares no deseados.
c) Servicios de almacenamiento en la nube (DropBox, Google Drive).

Sí. La aplicación utiliza el almacenamiento en tarjeta de memoria externa y en directorios que pueden compartirse con otras aplicaciones.

En las siguientes imágenes se muestra el código para el acceso a la memoria externa.

Figura 249. Timehop código acceso a la memoria externa.

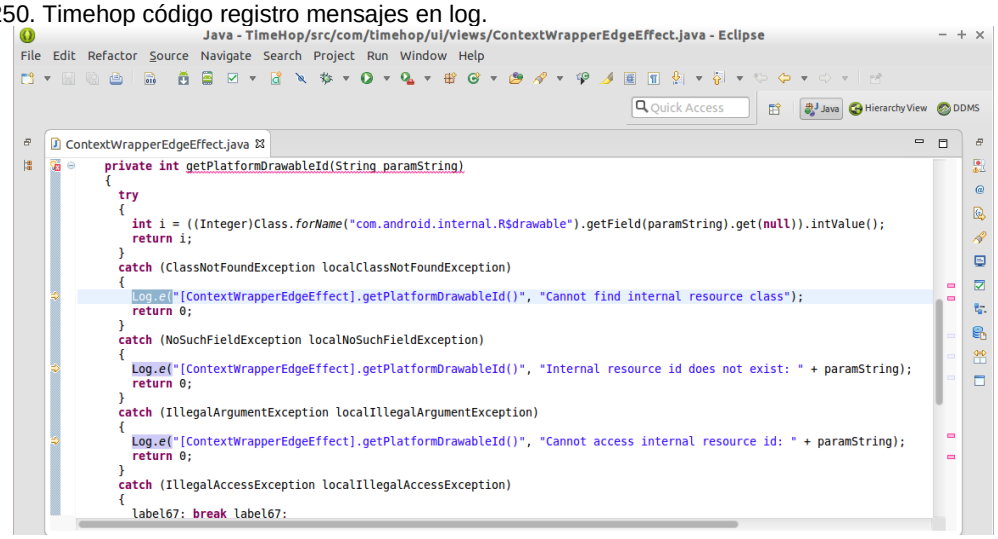


Fuente: Los Autores.

5- ¿La aplicación maneja un archivo de log? ¿Se puede acceder a información confidencial?

Si maneja archivo de log, la información registrada en el log no está cifrada.

Figura 250. Timehop código registro mensajes en log.



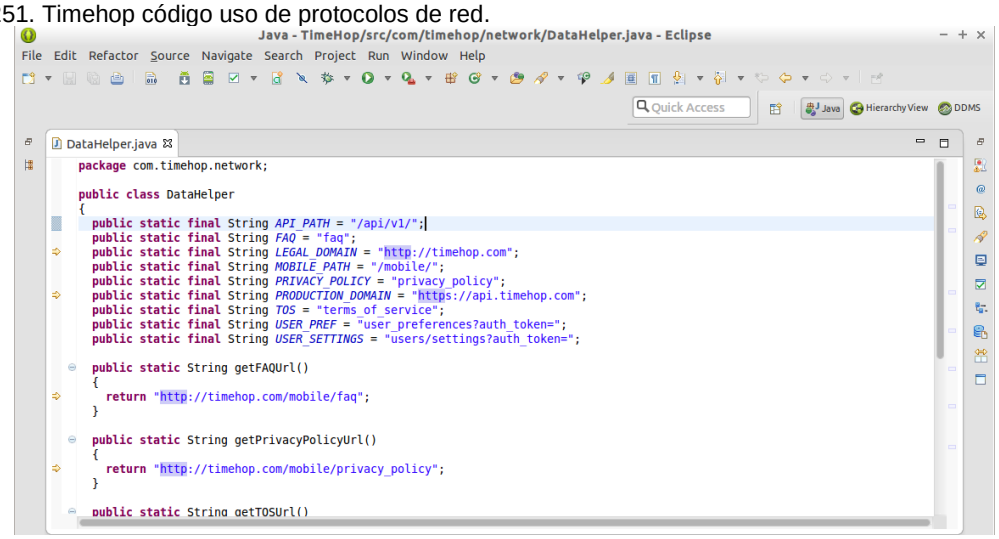
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

6- Identificar los Protocolos de red utilizados.

La aplicación utiliza los siguientes protocolos: http y https.

Figura 251. Timehop código uso de protocolos de red.



Fuente: Los Autores.

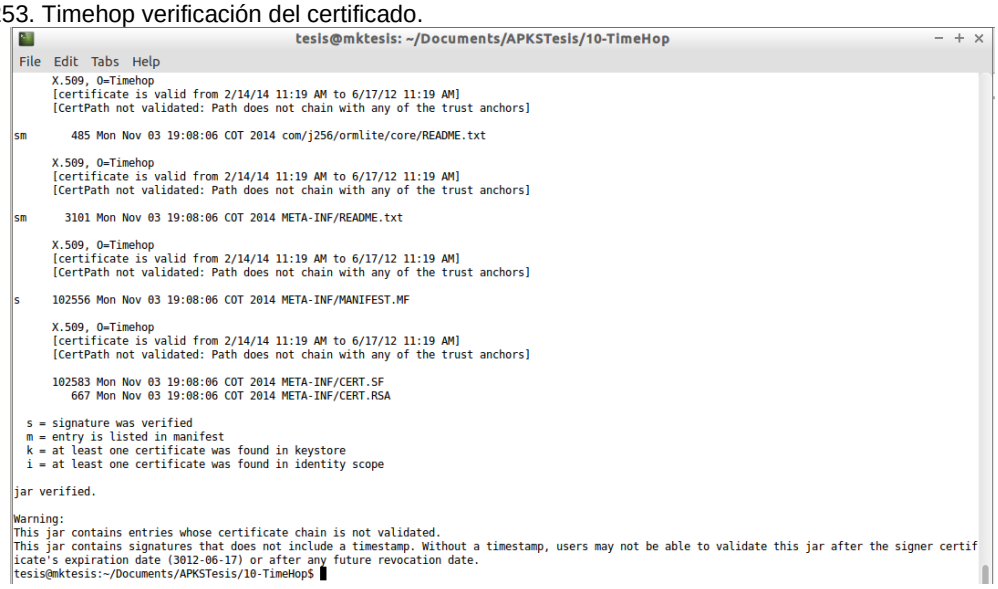
7- Identificar si la aplicación utiliza Certificados y determinar si valida la información de los mismos (caducidad, autoridad de certificación, validez, revocación, seguridad).

Se realiza verificación de la aplicación encontrándose que utiliza certificado, el cual se encuentra vigente y tiene una fecha de expiración ilimitada, lo que puede representar un riesgo de seguridad si un atacante logra suplantar el certificado.

Figura 252. Timehop información del certificado.

Fuente: Los Autores.

Figura 253. Timehop verificación del certificado.



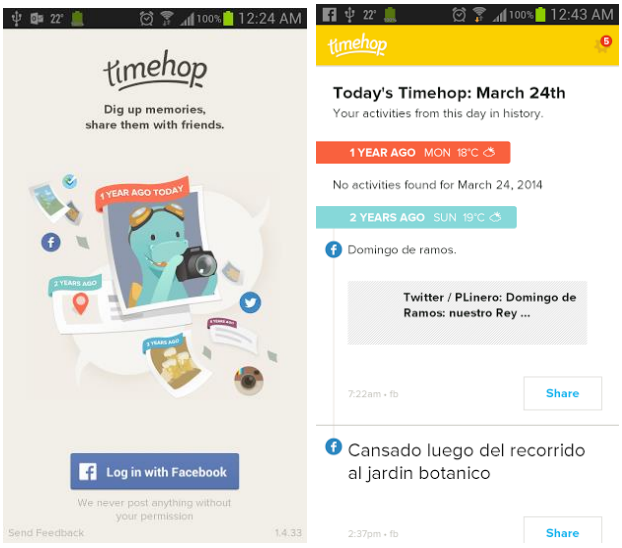
Fuente: Los Autores.

C- Análisis dinámico

1- Instalar, configurar y utilizar la aplicación.

Se instaló la aplicación, verificando su buen funcionamiento.

Figura 254. Timehop funcionamiento.



Fuente: Los Autores.

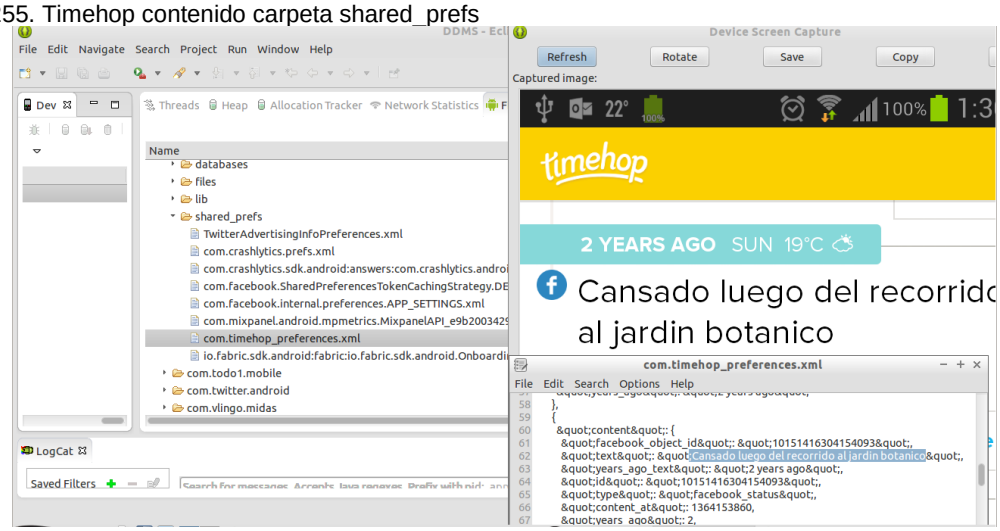
M2 Almacenamiento de datos inseguro

2- Determinar qué archivos y/o bases de datos fueron creadas por la aplicación.

La aplicación en el directorio “/data/data” crea las carpetas denominada “com.timehop” con las subcarpetas *cache*, *databases*, *files*, *lib* y *shared_prefs* con los correspondientes archivos.

Se observa en la carpeta “/shared_prefs”, el archivo com.t*imehop_prefs.xml* donde se identifica la información publicada pero no se observar datos sensibles.

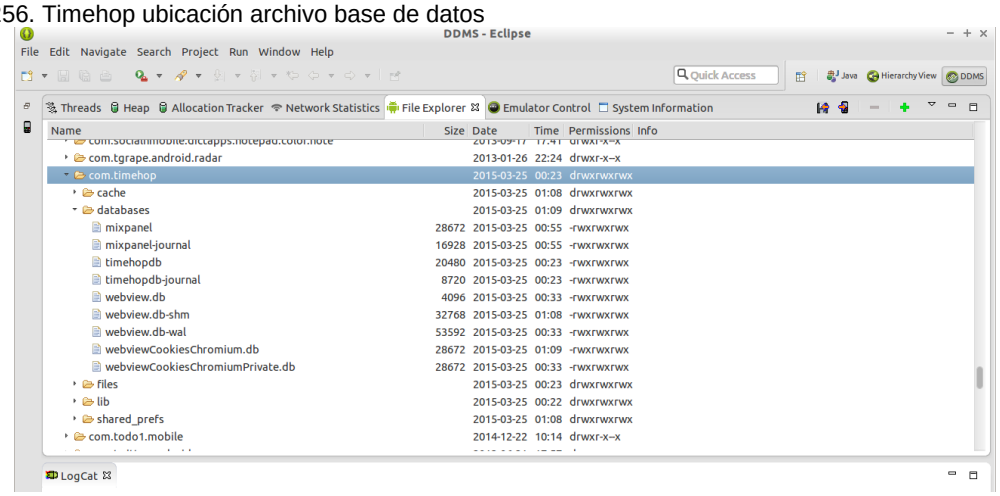
Figura 255. Timehop contenido carpeta shared_prefs



Fuente: Los Autores.

En la carpeta “databases” de la aplicación se puede observar la creación de las bases de datos “mixpanel”, “mixpanel-journal”, “timehopdb”, “timehopdb-journal”, “webviews.db” y “webviewsCookiesChromium.db” ”, las cuales no contienen información sensible de la aplicación.

Figura 256. Timehop ubicación archivo base de datos



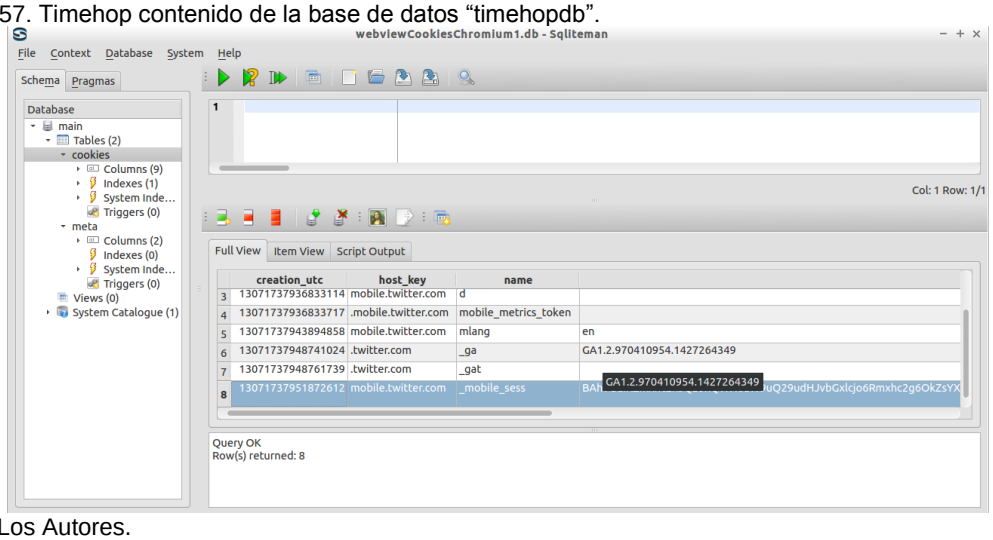
Fuente: Los Autores.

3- Revisar las bases de datos y/o archivos para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Revisada la base de datos “timehop” la información se encuentra cifrada.

La información encontrada no se considera sensible por cuanto no se observa almacenamiento de datos del usuario.

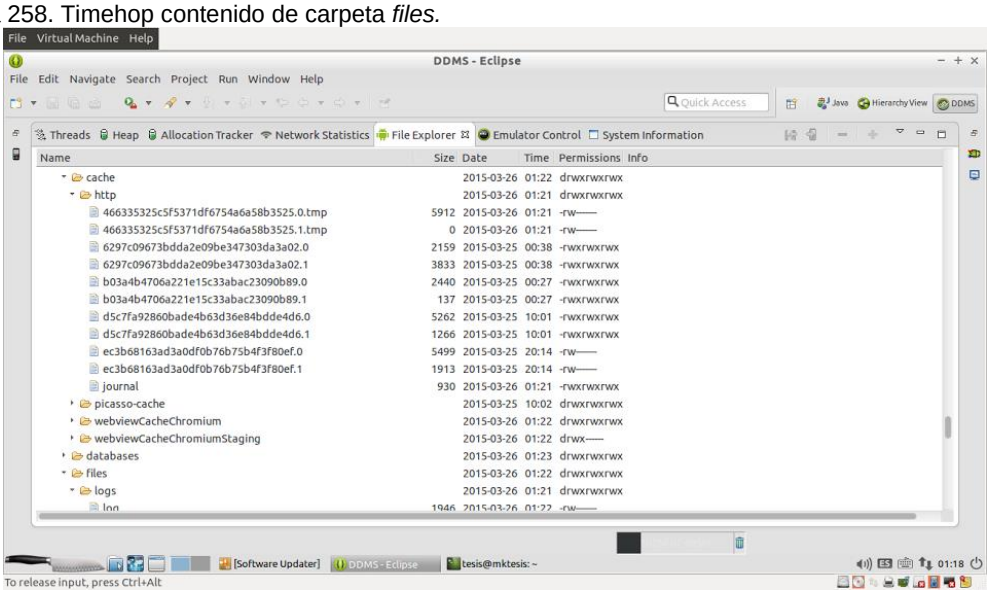
Figura 257. Timehop contenido de la base de datos “timehopdb”.



Fuente: Los Autores.

En la carpeta files no se encuentran archivos que manejen información sensible.

Figura 258. Timehop contenido de carpeta *files.*

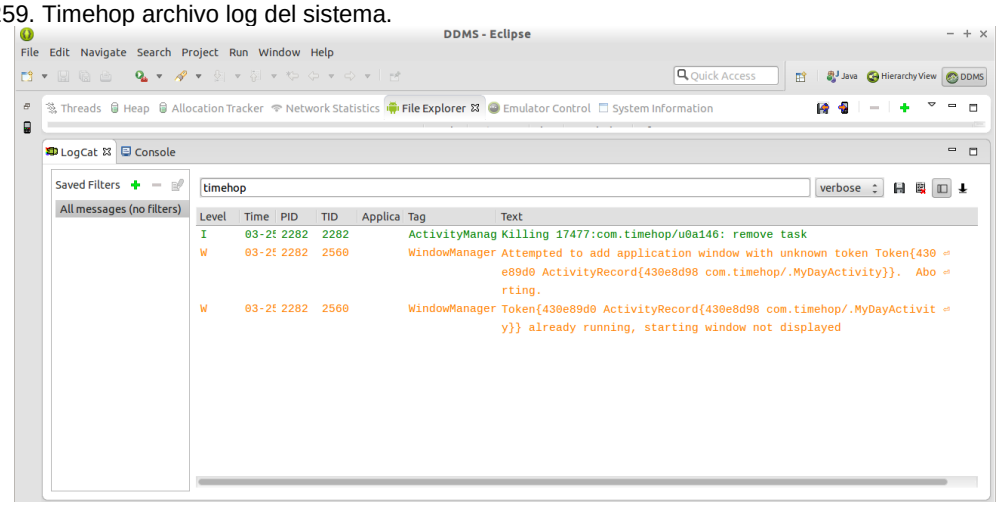


Fuente: Los Autores.

4- Revisar archivos de log para determinar qué datos se almacenan y si los datos sensibles están cifrados.

Analizando el archivo log se observa el almacenamiento de información cifrada, pero no es posible ver su contenido

Figura 259. Timehop archivo log del sistema.

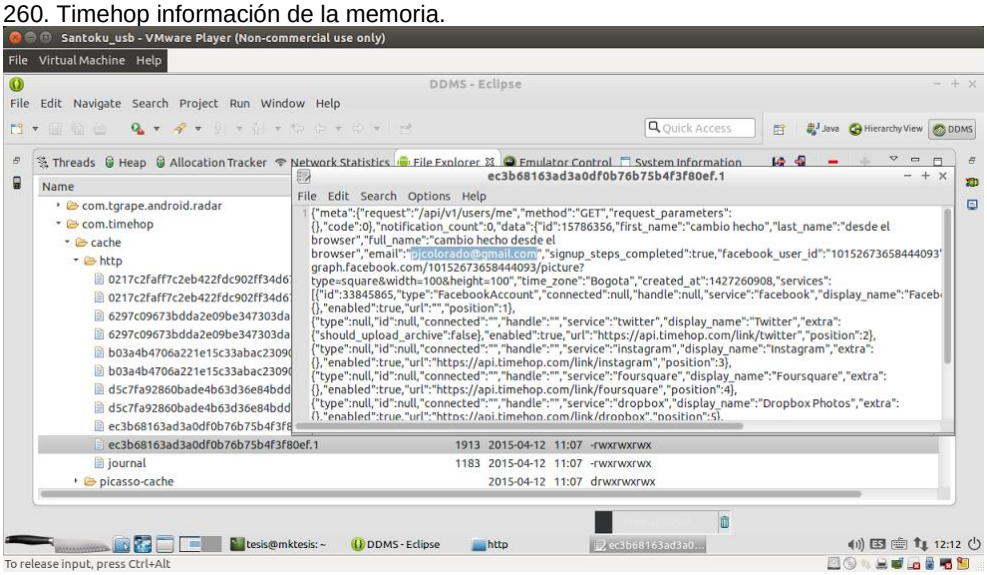


Fuente: Los Autores.

5- Analizar almacenamiento de datos en cache.

Se procedió a revisar la carpeta cache de la aplicación en donde se encontró almacenamiento de información sensible.

Figura 260. Timehop información de la memoria.

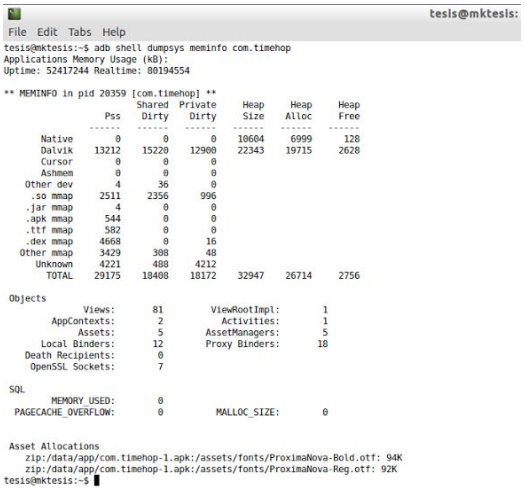


Fuente: Los Autores.

6- Determinar si la información sensible permanece en la memoria después de cerrar sesión en la aplicación.

Se realizaron comprobaciones de la memoria del dispositivo, una vez cerrada la aplicación se identifica que no permanece en memoria la información sensible de la misma.

Figura 261. Timehop información de la memoria.



```
tesis@mktesis:
File  Edit  Tabs  Help
tesis@mktesis:~$ adb shell dumpsys meminfo com.timehop
Applications Memory Usage (kB):
Uptime: 52417244 Realtime: 80194554

** MEMINFO in pid 20359 [com.timehop] **
                   Shared  Private     Heap     Heap     Heap
                     Pss   Dirty    Dirty     Size    Alloc     Free
                  ------  ------  ------  ------  ------  ------
       Native       0       0       0   10604    6999     128
       Dalvik   13212   15220   12900   22343   19715    2628
       Cursor       0       0       0
       Ashmem       0       0       0
    Other dev       4      36       0
     .so mmap    2511    2356     996
     .jar mmap      4       0       0
     .apk mmap    544       0       0
     .ttf mmap    582       0       0
     .dex mmap   4668       0      16
    Other mmap   3429     308      48
       Unknown   4221     488    4212
         TOTAL  29175   18408   18172   32947   26714    2756

 Objects
               Views:       81         ViewRootImpl:        1
         AppContexts:        2           Activities:        1
              Assets:        5        AssetManagers:        5
       Local Binders:       12        Proxy Binders:       18
    Death Recipients:        0
     OpenSSL Sockets:        7

 SQL
         MEMORY_USED:        0
  PAGECACHE_OVERFLOW:        0          MALLOC_SIZE:        0


 Asset Allocations
    zip:/data/app/com.timehop-1.apk:/assets/fonts/ProximaNova-Bold.otf: 94K
    zip:/data/app/com.timehop-1.apk:/assets/fonts/ProximaNova-Reg.otf: 92K
tesis@mktesis:~$
```

Fuente: Los Autores.

Figura 262. Timehop búsqueda de información en la memoria.

Fuente: Los Autores.

7- ¿Es posible obtener las claves de cifrado, credenciales, información de pago y otra información sensible mediante un volcado de memoria del dispositivo o de la aplicación?

Se realizó un volcado de memoria del dispositivo, realizando la búsqueda de información de la aplicación sin identificar información sensible.

Figura 263. Timehop búsqueda de información en el archivo hprof.

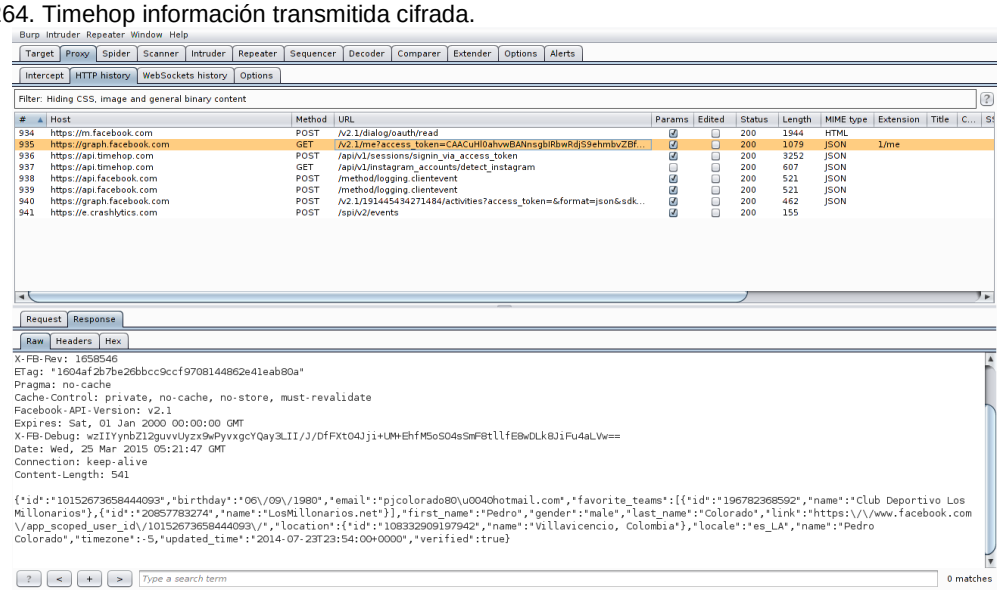
Fuente: Los Autores.

M3 Protección insuficiente en la capa de transporte

8- Analizar el tráfico de red para determinar si se envía información del usuario o datos sensibles no cifrados.

Realizada la interceptación del tráfico de la red se encontró que la aplicación utiliza protocolos HTTP y SSL en donde se envía en texto plano datos como el correo electrónico, la fecha de nacimiento, deportes favoritos, género y ubicación.

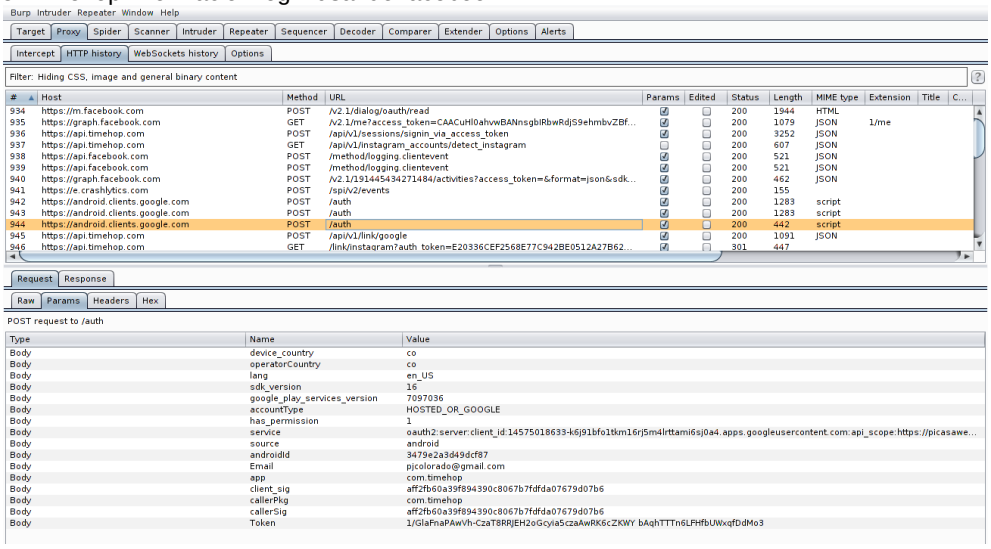
Figura 264. Timehop información transmitida cifrada.



Fuente: Los Autores.

Se realiza login con la cuenta de correo electrónico de facebook identificándose la información sensible del usuario como es el correo electrónico.

Figura 265. Timehop información login usando facebook.

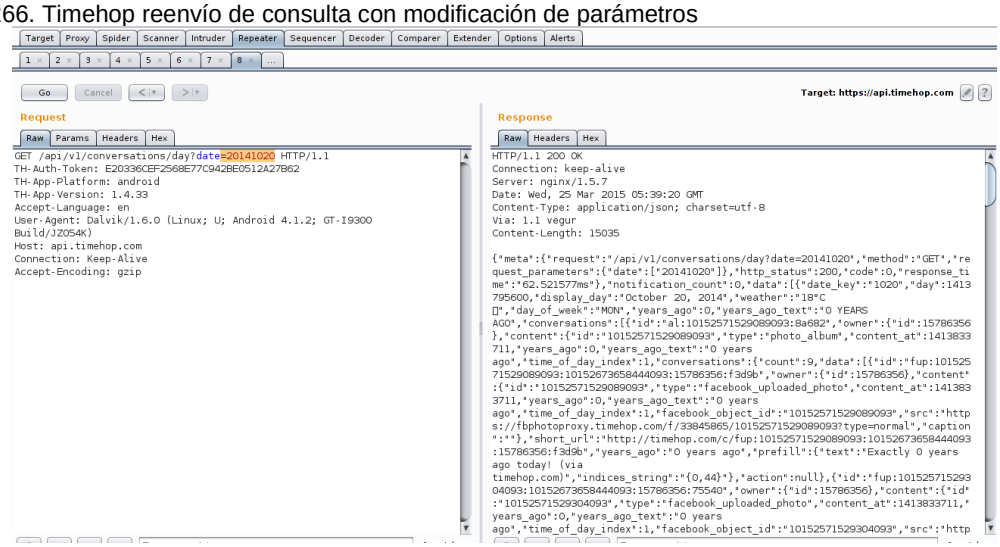


Fuente: Los Autores.

9- Determinar si se usan protocolos de comunicación de forma segura

Se procede al reenvío de una conexión modificando algunos parámetros y obteniendo respuesta positiva del servidor.

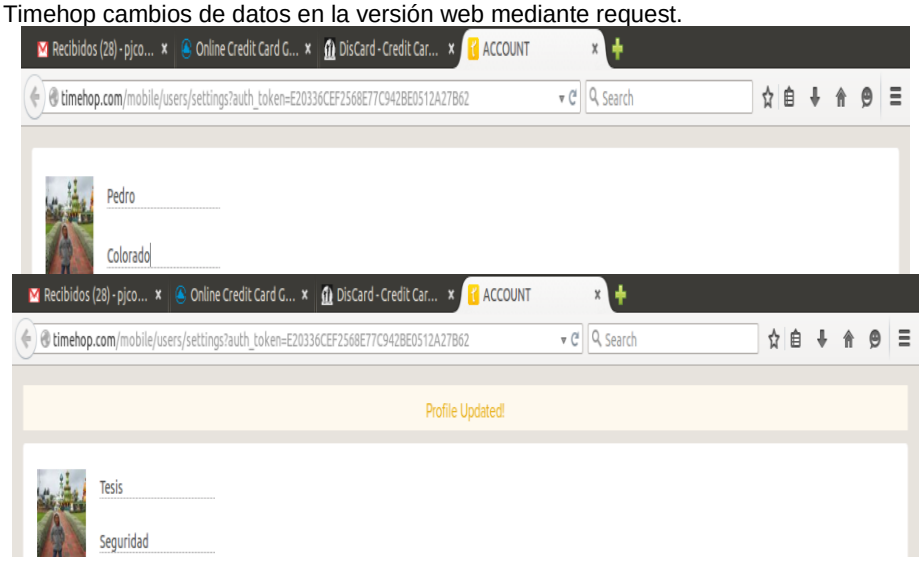
Figura 266. Timehop reenvío de consulta con modificación de parámetros



Fuente: Los Autores.

Se procede a realizar cambios en la versión web sobre los datos de contacto, permitiendo la modificación aun cuando la aplicación se encuentra ejecutando en el dispositivo móvil.

Figura 267. Timehop cambios de datos en la versión web mediante request.



Fuente: Los Autores.